From Approximate Factorization to Root Isolation with Application to Cylindrical Algebraic Decomposition

Kurt Mehlhorn and Michael Sagraloff and Pengming Wang*

January 22, 2013

Abstract

We present an algorithm for isolating the roots of an arbitrary complex polynomial p that also works for polynomials with multiple roots provided that the number k of distinct roots is given as part of the input. It outputs k pairwise disjoint disks each containing one of the distinct roots of p, and its multiplicity. The algorithm uses approximate factorization as a subroutine.

In addition, we apply the new root isolation algorithm to a recent algorithm for computing the topology of a real planar algebraic curve specified as the zero set of a bivariate integer polynomial and for isolating the real solutions of a bivariate polynomial system. For input polynomials of degree n and bit-size τ, we improve the currently best running time from $\tilde{O}(n^9\tau + n^8\tau^2)$ (deterministic) to $\tilde{O}(n^6 + n^5\tau)$ (randomized) for topology computation and from $\tilde{O}(n^8 + n^7\tau)$ (deterministic) to $\tilde{O}(n^6 + n^5\tau)$ (randomized) for solving bivariate systems.

1 Introduction

In the main part of this paper, we give an algorithm for isolating the roots of a polynomial $p(x) = \sum_{i=0}^{n} p_i x^i$ with arbitrary complex coefficients that works for polynomials with multiple roots, provided that the number k of distinct roots is given as part of the input. More precisely, let $z_1, \ldots, z_k$ be the distinct roots of p, let $m_i := \mathrm{mult}(z_i, p)$ be the *multiplicity* of z_i, and let $\sigma_i := \sigma(z_i, p) := \min_{j \neq i} |z_i - z_j|$ be the *separation* of z_i from the other roots of p. Then, the algorithm outputs isolating disks $\Delta_1 = \Delta(\tilde{z}_1, R_1)$ to $\Delta_k = \Delta(\tilde{z}_k, R_k)$ for the roots of p as well as the corresponding multiplicities m_1 to m_k. The radii satisfy $R_i < \frac{\sigma_i}{64n}$ for all i, thus the center $\tilde{z}_i$ of Δ_i approximates z_i to an error of less than $\frac{\sigma_i}{64n}$. If the number of distinct roots of p differs from k, we make no claims about termination and output.

The coefficients of p are provided by oracles. On input L, such an oracle essentially returns binary fraction approximations $\tilde{p}_i$ of the coefficients p_i such that $\left\| p - \sum_{i=0}^{n} \tilde{p}_i x^i \right\| \leq 2^{-L} \|p\|$. Here, $\|p\| := \|p\|_1 = |p_0| + \ldots + |p_n|$ denotes the *one-norm* of p. The details are given in Section 2.1.

The algorithm has a simple structure. We first use any algorithm (e.g. [4, 18, 16, 21]) for approximately factorizing the input polynomial. It is required that it can be run with different levels of precision and that, for any given integer b, it returns approximations $\hat{z}_1$ to $\hat{z}_n$ such that

$$\left\| p - p_n \prod_{1 \leq j \leq n} (x - \hat{z}_j) \right\| \leq 2^{-b} \|p\| . \tag{1}$$

In a second step, we partition the root approximations $\hat{z}_1$ to $\hat{z}_n$ into k clusters $C_1, \ldots, C_k$ based on geometric vicinity. We enclose each cluster C_i in a disk $D_i = \Delta(\tilde{z}_i, r_i)$ and make sure that the disks are pairwise disjoint and that the radii r_i are not "too small" compared to the pairwise distances of the centers $\tilde{z}_i$.[1] In a third step, we verify that the n-times enlarged disks $\Delta_i = \Delta(\tilde{z}_i, R_i) = \Delta(\tilde{z}_i, n \cdot r_i)$ are disjoint and that each of them contains exactly the same number of approximations as roots of p counted with multiplicity.[2] If the

*Max Planck Institute for Informatics, Germany

[1] This will turn out to be crucial to control the cost for the final verification step. For details, we refer to Sections 2.2.2 and 2.2.3.

[2] It seems to be artificial to consider enlarged disks Δ_i instead of the disks D_i. This is mainly due to the usage of Rouché's Theorem to verify roots.

clustering and the verification succeed, we return the disks $\Delta_1, \ldots, \Delta_k$ and the number of approximations $\hat{z} \in \{\hat{z}_1, \ldots, \hat{z}_n\}$ in the disk as the multiplicity of the root isolated by the disk. If either clustering or verification does not succeed, we repeat with a higher precision.

If Pan's algorithm [16] is used for the approximate factorization step, then the overall algorithm has bit complexity[3]

$$\tilde{O}\left(n^3 + n^2 \sum_{i=1}^{k} \log M(z_i) + n \sum_{i=1}^{k} \log\left(M(\sigma_i^{-m_i}) \cdot M(P_i^{-1})\right)\right) \tag{2}$$

where $P_i := \prod_{j \neq i}(z_i - z_j)^{m_j} = \frac{p^{(m_i)}(z_i)}{m_i! p_n}$, and $M(x) := \max(1, |x|)$. Observe that our algorithm is adaptive in a very strong sense, namely, the above bound directly depends on the actual multiplicities and the geometry (i.e. the actual modulus of the roots and their distances to each other) of the roots. There is also no dependency on the size or the type (i.e. whether they are rational, algebraic or transcendental) of the coefficients of p.

Our algorithm can also be used to further refine the isolating disks to a size of $2^{-\kappa}$ or less, where κ is a given integer. The bit complexity for the refinement is given by the bound in (2) plus an additional term $\tilde{O}(n \cdot \kappa \cdot \max_i m_i)$. In particular for square-free polynomials, the amortized cost per root and bit of precision is one.

For the benchmark problem of isolating all roots of a polynomial p with *integer* coefficients of absolute value bounded by 2^τ, the bound in (2) becomes $\tilde{O}(n^3 + n^2\tau)$. The bound for the refinement becomes $\tilde{O}(n^3 + n^2\tau + n\kappa)$, even if there exist multiple roots.

For a square-free integer polynomial p, we are aware of only one method [9, Theorem 3.1] that achieves a comparable complexity bound for the benchmark problem. That is, based on the gap theorem from Mahler, one can compute a theoretical worst case bound b_0 of size $\Theta(n\tau)$ with the property that if n points $\tilde{z}_j \in \mathbb{C}$ fulfill the inequality (1) for a $b \geq b_0$, then they approximate the corresponding roots z_j to an error less than $\sigma_j/(2n)$; cf. Lemma 4 for an adaptive version. Hence, for $b \geq b_0$, Pan's factorization algorithm also yields isolating disks for the roots of p using $\tilde{O}(n^2\tau)$ bit operations. Note that this approach achieves a good worst case complexity, however, for the price of running the factorization algorithm with $b = \Theta(n\tau)$, even if the roots are well conditioned. In contrast, our algorithm turns Pan's factorization algorithm into a highly adaptive method for isolating and approximating the roots of a general polynomial. Also, for general polynomials, there exist theoretical worst case bounds [18, Section 19] for the distance between the roots of p and corresponding approximations fulfilling (1). They are optimal for roots of multiplicity $\Omega(n)$ but they constitute strong overestimations if all roots have considerably smaller multiplicities. For the task of root approximation, the complexity of our method adapts to the highest occurring multiplicity (with bit complexity $\tilde{O}(n \max_i m_i \cdot \kappa)$ for κ dominating), whereas this is not given for the currently best method [16] (with complexity $\tilde{O}(n^2\kappa)$).

We would also like to remark that we are aware of only one previous root isolation algorithm [15] that can cope with multiple roots, however, with at most one multiple root. In addition, the number of distinct complex roots as well as the number of distinct real roots must be given as an additional input.

Finally, we aim to stress the importance of our root isolation method by applying it to the problems of computing the topology (in terms of a cylindrical algebraic decomposition) of a real planar algebraic curve specified as the zero set of an integer polynomial and of isolating the real solutions of a bivariate polynomial system. Both problems are well-studied [1, 11, 12, 19, 7, 5, 3, 8, 14]. More specifically, we apply our method to a recent algorithm [3] for computing the topology of a planar algebraic curve. This yields bounds on the *expected* number of bit operations which improve the currently best (which are both deterministic) bounds [8, 14] from $\tilde{O}(n^9\tau + n^8\tau^2)$ to $\tilde{O}(n^6 + n^5\tau)$ for topology computation and from $\tilde{O}(n^8 + n^7\tau)$ to $\tilde{O}(n^6 + n^5\tau)$ for solving bivariate systems.

[3] $\tilde{O}$ indicates that we omit logarithmic factors.

2 Root Approximation

2.1 Setting and Basic Properties

We consider a polynomial

$$p(x) = p_n x^n + \ldots + p_0 \in \mathbb{C}[x] \tag{3}$$

of degree $n \geq 2$, where $p_n \neq 0$. We fix the following notations:

- $M(x) := \max(1, |x|)$, for $x \in \mathbb{R}$,
- τ_p denotes the minimal non-negative integer with $\frac{|p_i|}{|p_n|} \leq 2^{\tau_p}$ for all $i = 0, \ldots, n-1$,
- $\|p\| := \|p\|_1 := |p_0| + \ldots + |p_n|$ denotes the 1-norm of p,
- $z_1, \ldots, z_k \in \mathbb{C}$ are the distinct complex roots of p, with $k \leq n$,
- $m_i := \mathrm{mult}(z_i, p)$ is the multiplicity of z_i,
- $\sigma_i := \sigma(z_i, p) := \min_{j \neq i} |z_i - z_j|$ is the *separation* of z_i,
- $\Gamma_p := M(\max_i \log |z_i|)$ the *logarithmic root bound* of p,
- $\mathrm{Mea}(p) = |p_n| \cdot \prod_i M(z_i)^{m_i}$ the *Mahler Measure* of p.

The quantities τ_p, Γ_p, $|p_n|$ and $\mathrm{Mea}(p)$ are closely related.

Lemma 1. *$\Gamma_p \leq 1 + \tau_p$ and $\tau_p - n - 1 \leq \log \frac{\mathrm{Mea}(p)}{|p_n|} \leq n\Gamma_p$.*

Proof. By Cauchy's root bound $\max_i |z_i| \leq 1 + \max_i |p_i| / |p_n|$, and thus $\max_i \log |z_i| \leq 1 + \tau_p$. Since $\tau_p \geq 0$, by definition, we have $\Gamma_p \leq 1 + \tau_p$. The i-th coefficient of p is smaller than or equal to $\binom{n}{i} \mathrm{Mea}(p) \leq 2^n \mathrm{Mea}(p) \leq 2^{n(\Gamma_p + 1)}$. Thus, from the definition of τ_p, either $\tau_p = 0$ or $2^n \frac{\mathrm{Mea}(p)}{|p_n|} \geq \max_i \frac{|p_i|}{|p_n|} \geq 2^{\tau_p - 1}$ □

We assume the existence of an oracle which provides arbitrary good approximations of the polynomial p. Let $L \geq 1$ be an integer. We call a polynomial $\tilde{p} = \tilde{p}_n x^n + \ldots + \tilde{p}_0$, with $\tilde{p}_i = s_i \cdot 2^{-\ell}$ and s_i, $\ell \in \mathbb{Z}$, an *approximation of precision L* of p if $|\tilde{p}_i - p_i| \leq 2^{-L - \log(n+1)} \|p\|$, $\ell \leq L + \lceil \log(n+1) \rceil - \lfloor \log \|p\| \rfloor$, and $\log |s_i| \leq L + \lceil \log(n+1) \rceil + 1$ for all i. When considering p_i as infinite bitstring $p_i = \mathrm{sgn}(p_i) \cdot \sum_{k=-\infty}^{+\infty} b_k 2^k$, $b_k \in \{0, 1\}$, then we can obtain $\tilde{p}_i$ from the partial string which starts at index $k_1 = \lfloor \log \|p\| \rfloor$ and ends at index $k_2 = \lfloor \log \|p\| \rfloor - L - \lceil \log(n+1) \rceil$, that is, $s_i := 2^l \cdot \mathrm{sgn}(p_i) \cdot \sum_{k=k_2}^{k_1} b_k 2^k$, and $l = L + \lceil \log(n+1) \rceil - \lfloor \log \|p\| \rfloor$. *We assume that we can ask for an approximation of precision L of p at cost $O(n(L + \log n)) = \tilde{O}(nL)$.* This is the cost of reading the approximation of precision L. The next Lemma states some elementary properties of approximations of precision L.

Lemma 2. *Let $\tilde{p}$ be an approximation of precision L of p and $L \geq 1$.*

- $\|\tilde{p}\| / 2 \leq \|p\| \leq 2 \|\tilde{p}\|$.
- *If $L \geq \tau_p + 4$, then $2^{-L - \log(n+1)} \|\tilde{p}\| \leq |\tilde{p}_n| / 4$.*
- *If $2^{-L - \log(n+1)} \|\tilde{p}\| \leq |\tilde{p}_n| / 4$, then $|\tilde{p}_n| / 2 \leq |p_n| \leq 2 |\tilde{p}_n|$.*

Proof. From the inequality $|\|\tilde{p}\| - \|p\|| \leq \sum_i |\tilde{p}_i - p_i|$, we conclude that $|\|\tilde{p}\| - \|p\|| \leq 2^{-L} \|p\| \leq \|p\| / 2$. This establishes the first claim. For the second claim, we observe that

$$|\tilde{p}_n - p_n| \leq 2^{-L - \log(n+1)} \|p\| \leq 2^{-L + \tau_p} |p_n| \leq |p_n| / 16.$$

Thus, $|\tilde{p}_n| \geq |p_n| / 2$ and

$$2^{-L - \log(n+1)} \|\tilde{p}\| \leq 2 \cdot 2^{-L - \log(n+1)} \|p\| \leq |p_n| / 8 \leq |\tilde{p}_n| / 4.$$

The third claim follows from

$$|\tilde{p}_n - p_n| \le 2^{-L-\log(n+1)} \|p\| \le 2 \cdot 2^{-L-\log(n+1)} \|\tilde{p}\| \le |\tilde{p}_n|/2.$$

□

Lemma 2 suggests an efficient method for estimating p_n. We ask for approximations $\tilde{p}$ of precision L of p for $L = 1, 2, 4, \ldots$ until the inequality $2^{-L-\log(n+1)} \|\tilde{p}\| \le |\tilde{p}_n|/4$ holds. Then, $|\tilde{p}_n|/2 \le |p_n| \le 2|\tilde{p}_n|$ by part 3 of the Lemma. Also $L \le 2(\tau_p + 4)$ by part 2 of the above Lemma. The cost is $\tilde{O}(n\tau_p) = \tilde{O}(n^2\Gamma_p)$ bit operations, where we used the upper bound for τ_p from Lemma 1. Observe that this bound depends only on the geometry of the roots (i.e. the actual root bound Γ_p) and the degree but not (directly) on the size of the coefficients of p. We next show that a "good" integer approximation Γ of Γ_p can also be computed with $\tilde{O}(n^2\Gamma_p)$ bit operations.

Theorem 1. *An integer $\Gamma \in \mathbb{N}$ with*

$$\Gamma_p \le \Gamma < 8\log n + \Gamma_p \tag{4}$$

can be computed with $\tilde{O}(n^2\Gamma_p)$ bit operations. The computation uses an approximation of precision L of p with $L = O(n\Gamma_p)$.

Proof. We first compute an approximation $\tilde{p}$ of precision L of p with $|\tilde{p}_n|/2 \le |p_n| \le 2 \cdot |\tilde{p}_n|$ as described above. Let $\kappa := 2^{\lfloor \log|\tilde{p}_n| \rfloor - 1}$. Then $\kappa \le |\tilde{p}_n|/2 \le |p_n|$ and $\kappa \ge 2^{\lfloor \log|p_n/2| \rfloor - 1} \ge |p_n|/8$. Consider the scaled polynomial $q := p/\kappa$. Its leading coefficient lies between 1 and 8. Moreover, $\Gamma_q = \Gamma_p$, $\tau_q = \tau_p$, and an arbitrary approximation of precision L of p also yields an approximation of precision L of q since division by λ is just a shift by $\log\kappa$ bits and $\|q\| = \|p\|/\kappa$. Hence, we may assume that $1 \le |p_n| \le 8$.

Consider the *Cauchy polynomial*

$$\bar{p}(x) := |p_n|x^n - \sum_{i=0}^{n-1} |p_i|x^i$$

of p. Then, according to [6, Proposition 2.51], $\bar{p}$ has a unique positive real root $\xi \in \mathbb{R}^+$, and the following inequality holds:

$$\max_i |z_i| \le \xi < \frac{n}{\ln 2} \cdot \max_i |z_i| < 2n \cdot \max_i |z_i|.$$

It follows that $\bar{p}(x) > 0$ for all $x \ge \xi$ and $\bar{p}(x) < 0$ for all $x < \xi$. Furthermore, since $\bar{p}$ coincides with its own Cauchy polynomial, each complex root of $\bar{p}$ has absolute value less than or equal to $|\xi|$. Let k_0 be the smallest non-negative integer k with $\bar{p}(2^k) > 0$ (which is equal to the smallest k with $2^k > \xi$). Our goal is to compute an integer Γ with $k_0 \le \Gamma \le k_0 + 1$. Namely, if Γ fulfills the latter inequality, then $M(\max_i |z_i|) \le M(\xi) \le 2^\Gamma < 4M(\xi) < 8n \cdot M(\max_i |z_i|)$, and thus Γ fulfills inequality (4). In order to compute a Γ with $k_0 \le \Gamma \le k_0 + 1$, we use exponential and binary search (try $k = 1, 2, 4, 8, \ldots$ until $\bar{p}(2^k) > 0$ and, then, perform binary search on the interval $k/2$ to k) and approximate evaluation of $\bar{p}$ at the points 2^k: More precisely, we evaluate $\bar{p}(2^k)$ using interval arithmetic with a precision ρ (using fixed point arithmetic) which guarantees that the width w of $\mathfrak{B}(\bar{p}(2^k), \rho)$ is smaller than 1, where $\mathfrak{B}(E, \rho)$ is the interval obtained by evaluating a polynomial expression E via interval arithmetic with precision ρ for the basic arithmetic operations; see [13, Section 4] for details. We use [13, Lemma 3] to estimate the cost for each such evaluation: Since $\bar{p}$ has coefficients of size less than $2^{\tau_p}|p_n| < 2^{\tau_p+3}$, we have to choose ρ such that

$$2^{-\rho+2}(n+1)^2 2^{\tau_p+3+nk} < 1$$

in order to ensure that $w < 1$. Hence, ρ is bounded by $O(\tau_p + nk)$ and, thus, each interval evaluation needs $\tilde{O}(n(\tau_p + nk))$ bit operations. We now use exponential plus binary search to find the smallest k such that $\mathfrak{B}(\bar{p}(2^k), \rho)$ contains only positive values. The following argument then shows that $k_0 \le k \le k_0 + 1$: Obviously, we must have $k \ge k_0$ since $\bar{p}(2^k) < 0$ and $\bar{p}(2^k) \in \mathfrak{B}(\bar{p}(2^k), \rho)$ for all $k < k_0$. Furthermore, the point $x = 2^{k_0+1}$ has distance more than 1 to each of the roots of $\bar{p}$, and thus $|\bar{p}(2^{k_0+1})| \ge |p_n| \ge 1$. Hence, it follows that $\mathfrak{B}(\bar{p}(2^{k_0}+1), \rho)$ contains only positive values. For the search, we need

$$O(\log k_0) = O(\log\log\xi) = O(\log(\log n + \Gamma_p))$$

iterations, and the cost for each of these iterations is bounded by $\tilde{O}(n(\tau_p + nk_0)) = \tilde{O}(n^2\Gamma_p)$ bit operations. □

2.2 Algorithm

We present an algorithm for isolating the roots of a polynomial $p(x) = \sum_{i=0}^{n} p_i x^i = p_n \prod_{i=1}^{k} (x - z_i)^{m_i}$, where the coefficients p_i are given as described in the previous section. We may assume $k > 1$; the problem is trivial otherwise. If $k = 1$, $-p_{n-1}/(np_n)$ is the root of multiplicity n. The algorithm uses some polynomial factorization algorithm to produce approximations for the roots $z_1, \ldots, z_k$, and then performs a clustering and certification step to verify that the candidates are of high enough quality. For concreteness, we pick Pan's factorization algorithm [16] for the factorization step, which also currently offers the best worst case bit complexity. If the candidates do not pass the verification step, we reapply the factorization algorithm with a higher precision. Given a polynomial p with $|z_i| \leq 1$ for $1 \leq i \leq k$, and a positive integer b denoting the desired precision, the factorization algorithm computes n root approximations $\hat{z}_1, \ldots, \hat{z}_n$. The quality of approximation and the bit complexity are as follows:

Theorem 2 (Pan [16])**.** *Suppose that $|z_i| \leq 1$ for $1 \leq i \leq k$. For any positive integer $b \geq n \log n$, complex numbers $\hat{z}_1, \ldots, \hat{z}_n$ can be computed such that they satisfy*

$$\left\| p - p_n \prod_{i=1}^{n} (x - \hat{z}_i) \right\| \leq 2^{-b} \|p\|$$

using $\tilde{O}(n)$ operations performed with the precision of $O(b)$ bits (or $\tilde{O}(bn)$ bit-operations). We write $\hat{p} := p_n \prod_{i=1}^{n} (x - \hat{z}_i)$. The algorithm returns the real and imaginary part of the $\hat{z}_i$'s as dyadic fractions of the form $A \cdot 2^{-B}$ with $A \in \mathbb{Z}$, $B \in \mathbb{N}$ and $B = O(b)$. All fractions have the same denominator.

The parameter b controls the quality of the resulting approximations. Note that Pan's algorithm requires all roots of the input polynomial to lie within the unit disk $\Delta(0,1)$. Hence, in order to apply the above result to our input polynomial, we first scale p such that the roots come to lie in the unit disk. That is, we compute a Γ as in Theorem 1, and then consider the polynomial $f(x) := p(s \cdot x) = \sum_{i=0}^{n} f_i x^i$ with $s := 2^{\Gamma}$. Then, $f(x)$ has roots $\xi_i = z_i/s \in \Delta(0,1)$, and thus we can use Pan's Algorithm with $b' := n\Gamma + b$ to compute an approximate factorization $\hat{f}(x) := \sum_{i=0}^{n} \hat{f}_i x^i := f_n \prod_{i=1}^{n} (x - \hat{\xi}_i)$ such that $\left\| f - \hat{f} \right\| < 2^{-b'} \|f\|$. Let $\hat{z}_i := s \cdot \hat{\xi}_i$ for all i and $\hat{p}(x) := p_n \cdot \prod_{i=1}^{n} (x - \hat{z}_i) = \hat{f}(x/s) = \sum_{i=0}^{n} \hat{f}_i / s^i x^i$. Then

$$\begin{aligned} \|\hat{p} - p\| &= \sum_{i=0}^{n} |f_i/s^i - \hat{f}_i/s^i| \leq \sum_{i=0}^{n} |f_i - \hat{f}_i| \leq 2^{-b'} \sum_{i=0}^{n} |f_i| \\ &\leq 2^{-b'} s^n \sum_{i=0}^{n} |f_i/s^i| = 2^{-b} \|p\| . \end{aligned}$$

For the factorization of f, we need an approximation of precision b' of f, and thus an approximation of precision L of p with $L = O(nS + b) = \tilde{O}(n\Gamma_p + b)$. The total cost is $\tilde{O}(n^2\Gamma + nb)$ bit operations. We summarize in:

Corollary 1. *For an arbitrary polynomial p and an integer $b \geq n \log n$, complex numbers $\hat{z}_1, \ldots, \hat{z}_n$ can be computed such that*

$$\left\| p - p_n \prod_{i=1}^{n} (x - \hat{z}_i) \right\| \leq 2^{-b} \|p\|$$

using $\tilde{O}(n^2\Gamma + bn)$ bit-operations. We write $\hat{p} := p_n \prod_{i=1}^{n} (x - \hat{z}_i)$. The algorithm returns the real and imaginary part of the $\hat{z}_i$'s as dyadic fractions of the form $A \cdot 2^{-B}$ with $A \in \mathbb{Z}$, $B \in \mathbb{N}$ and $B = O(b + n\Gamma_p)$. All fractions have the same denominator.

We now examine how far the approximations $\hat{z}_1, \ldots, \hat{z}_n$ can deviate from the actual roots for a given value of b. Let $\Delta(z,r)$ be the disk with center z and radius r and let $\mathrm{bd}\Delta(z,r)$ be its boundary. We further define $P_i := \prod_{j \neq i} (z_i - z_j)^{m_j}$. Then, $p^{(m_i)}(z_i) = m_i! p_n P_i$.

Lemma 3. *If $r \leq \sigma_i/n$, then*

$$|p(x)| > r^{m_i} |p_n P_i| / 4$$

for all x on the boundary of $\Delta(z_i, r)$.

Proof. We have

$$\begin{aligned}|p(x)| &= |p_n|\,|x-z_i|^{m_i}\prod_{j\neq i}|x-z_j|^{m_j}\\ &\geq |p_n|\,|x-z_i|^{m_i}\prod_{j\neq i}|z_i-z_j|^{m_j}\cdot(1-|x-z_i|/|z_i-z_j|)^{m_j}\\ &\geq r^{m_i}(1-1/n)^{n-m_i}\,|p_n|\prod_{j\neq i}|z_i-z_j|^{m_j} > r^{m_i}|p_nP_i|/4.\end{aligned}$$

□

Based on the above Lemma, we can now use Rouché's theorem [4,5] to show that, for sufficiently large b, the disk $\Delta(z_i, 2^{-b/(2m_i)})$ contains exactly m_i root approximations.

Lemma 4. *Let $\hat{p}$ be such that $\|p-\hat{p}\| \leq 2^{-b}\|p\|$. If*

$$b \geq \max(8n, n\log(n)),\ \textit{and } b \textit{ is a power of two} \tag{5}$$

$$2^{-b/(2m_i)} \leq 1/(2n^2), \tag{6}$$

$$2^{-b/(2m_i)} \leq \sigma_i/(2n),\ \textit{and} \tag{7}$$

$$2^{-b/2} \leq \frac{|P_i|}{16(n+1)2^{\tau_p}M(z_i)^n} \tag{8}$$

for all i, the disk $\Delta(z_i, 2^{-b/(2m_i)})$ contains exactly m_i root approximations. For $i \neq j$, let $\hat{z}_i$ and $\hat{z}_j$ be arbitrary approximations in the disks $\Delta(z_i, 2^{-b/(2m_i)})$ and $\Delta(z_j, 2^{-b/(2m_j)})$, respectively. Then,

$$(1-\frac{1}{n})\left|z_i-z_j\right| \leq \left|\hat{z}_i-\hat{z}_j\right| \leq (1+\frac{1}{n})\left|z_i-z_j\right|.$$

Proof. Let

$$\delta_i := \left(16\cdot(n+1)\cdot 2^{-b}2^{\tau_p}|P_i|^{-1}M(z_i)^n\right)^{1/m_i}.$$

It is easy to verify that $\delta_i \leq 2^{-b/(2m_i)} \leq \min(1,\sigma_i)/(2n)$. The first inequality follows from (8) and the second inequality follows from (6) and (7). We will show that $\Delta(z_i,\delta_i)$ contains m_i approximations. To this end, is suffices to show that $|(p-\hat{p})(x)| < |p(x)|$ for all x on the boundary of $\Delta(z_i,\delta_i)$. Then, Rouché's theorem guarantees that $\Delta(z_i,\delta_i)$ contains the same number of roots of p and $\hat{p}$ counted with multiplicity. Since z_i is of multiplicity m_i and $\delta_i < \sigma_i/n$, the disk contains exactly m_i roots of p counted with multiplicity. We have (note that $|x| \leq (1+1/(2n^2))\cdot M(z_i)$ for $x \in \mathrm{bd}\Delta(z_i,\delta_i)$)

$$\begin{aligned}|(p-\hat{p})(x)| &\leq \|p-\hat{p}\|\cdot|x|^n < 2^{-b}\|p\|\,|x|^n\\ &\leq 2^{-b}\|p\|\cdot(1+1/(2n^2))^n\cdot M(z_i)^n\\ &\leq 4\cdot 2^{-b}\cdot 2^{\tau_p}|p_n|\cdot(n+1)\cdot M(z_i)^n\\ &\leq \delta_i^{m_i}|p_nP_i|/4 < |p(x)|,\end{aligned}$$

where the inequality in line three follows from $\|p\| \leq (n+1)|p_n|2^{\tau_p}$, the first one in line four follows from the definition of δ_i, and the last inequality follows from Lemma 3. It follows that $\Delta(z_i, 2^{-b/(2m_i)})$ contains exactly m_i approximations. Furthermore, since $\delta_i \leq \sigma_i/(2n)$ for all i, the disks $\Delta(z_i,\delta_i)$, $1 \leq i \leq k$, are pairwise disjoint.

For the second claim, we observe that $|\hat{z}_\ell - z_\ell| \leq 2^{-b/(2m_\ell)} \leq \sigma_\ell/(2n) \leq \left|z_i-z_j\right|/(2n)$ for $\ell = i, j$ and hence $|\hat{z}_i - z_i| + \left|\hat{z}_j - z_j\right| \leq \left|z_i-z_j\right|/n$. The claim now follows from the triangle inequality. □

[4] Rouché's theorem states that if f and g are holomorphic functions with $|(f-g)(x)| < |f(x)|$ for all points x on the boundary of some disk Δ, then f and g have the same number of zeros (counted with multiplicity) in Δ.

[5] The reader may wonder why we are not using Gershgorin circles to cluster and verify root approximations. The problem is that very good approximations of a multiple root generate very large Gershgorin circles which may then fail to isolate the approximations. Indeed, if two approximations are equal, the corresponding disks have infinite radius.

We have now established that the disks $\Delta(z_i, 2^{-b/(2m_i)})$, $1 \le i \le k$, are pairwise disjoint and that the i-th disk contains exactly m_i root approximations provided that b satisfies (5) to (8). Unfortunately, the conditions on b are stated in terms of the quantities m_i, σ_i and $|P_i|$ which we do know. Also, we do not know the center z_i. In the remainder of the section, we will show how to cluster root approximations and to certify them. We will need the following more stringent properties for the clustering and certification step.

$$2^{-b/(2m_i)} < \min\left(\left(\frac{\sigma_i}{4n}\right)^8, \frac{\sigma_i}{1024n^2}\right) \tag{9}$$

$$2^{-b/8} < \min(1/16, |P_i| / ((n+1) \cdot 2^{2n\Gamma_p + 8n})) \tag{10}$$

Let b_0 be the smallest integer satisfying (5) to (10) for all i. Then,

$$b_0 = O(n \log n + n\Gamma_p + \max_i(m_i \log M(\sigma_i^{-1}) + \log M(P_i^{-1})).$$

We next provide a high-level description of our algorithm to isolate the roots of p. The details of the clustering step and the certification step are then given in Sections 2.2.2 and 2.2.3, respectively.

2.2.1 Overview of the Algorithm

On input p and the number k of distinct roots, the algorithm outputs isolating disks $\Delta_i = \Delta(\tilde{z}_i, R_i)$ for the roots of p as well as the corresponding multiplicities m_i. The radii satisfy $R_i < \sigma_i/(64n)$.

The algorithm uses the factorization step with an increasing precision until the result can be certified. If either the clustering step or the certification step fails, we simply double the precision. There are a couple of technical safeguards to ensure that we do not waste time on iterations with an insufficiently large precision (Steps 2, 5, and 6); also recall that we need to scale our initial polynomial.

1. Compute the bound 2^Γ for the modulus of all roots of p, where Γ fulfills Inequality (4). According to Theorem 1, this can be done with $\tilde{O}(n^2\Gamma_p)$ bit operations.

2. Compute a 2-approximation $\lambda = 2^{l_\lambda}$, $l_\lambda \in \mathbb{Z}$, of $\|p\| / |p_n|$. According to Lemma 2 and the subsequent remarks, we can compute λ with $\tilde{O}(n^2\Gamma_p)$ bit operations.

3. Scale p, that is, $f(x) := p(s \cdot x)$, with $s := 2^\Gamma$, to ensure that the roots $\xi_i = z_i/S$, $i = 1, \ldots, k$, of f are contained in the unit disk. Let b be the smallest integer satisfying (5)

4. Run Pan's algorithm on input f with parameter $b' := b + n\Gamma$ to produce approximations $\hat{\xi}_1, \ldots, \hat{\xi}_n$ for the roots of f. Then, $\hat{z}_i := s \cdot \hat{\xi}_i$ are approximations of the roots of p, and $\|\hat{p} - p\| < 2^{-b} \|p\|$, where $\hat{p}(x) := p_n \prod_{i=1}^n (x - \hat{z}_i)$.

5. If there is a $\hat{z}_i$ with $\hat{z}_i \ge 2^{\Gamma+1}$, return to Step 4 with $b := 2b$.

6. If $\prod_{i=1}^n M(\hat{z}_i) > 8\lambda$, return to Step 4 with $b := 2b$.

7. Partition $\hat{z}_1, \ldots, \hat{z}_n$ into k clusters $C_1, \ldots, C_k$. Compute (well separated) enclosing disks $D_1, \ldots, D_k$ for the clusters. If the clustering fails to find k clusters and corresponding disks, return to Step 4 with $b := 2b$.

8. For each i, let Δ_i denote the disk with the same center as D_i but with an n-times larger radius. We now verify the existence of $|C_i|$ roots (counted with multiplicity) of p in Δ_i. If the verification fails, return to Step 4 with $b := 2b$.

9. If the verification succeeds, output the disks Δ_i (in Step 7, we guarantee that the disks Δ_i are pairwise disjoint) and report the number $|C_i|$ of root approximations $\hat{z} \in \{\hat{z}_1, \ldots, \hat{z}_n\}$ contained in the disks as the corresponding multiplicities.

Note that Steps 5 and 6 ensure that $\log M(\hat{z}_i) = O(\Gamma_p + \log n)$ for all i, and $\log \prod_{i=1}^n M(\hat{z}_i) = O(\log(\|p\| / |p_n|)) = O(\log n + \tau_p) = \tilde{O}(n\Gamma_p)$. The following Lemma guarantees that the algorithm passes these steps if $b \ge b_0$.

Lemma 5. *For any $b \geq b_0$, it holds that $|\hat{z}_i| < 2^{\Gamma+1}$ for all i, and $\prod_{i=1}^{n} M(\hat{z}_i) < 8\lambda$.*

Proof. In the proof of Lemma 4, we have already shown that $|\hat{z}_i| \leq (1+1/(2n^2)) \cdot M(z_i)$ for all i. Hence, it follows that $|\hat{z}_i| \leq (1+1/(2n^2)) \cdot 2^{\Gamma_p} < 2 \cdot 2^{\Gamma_p} \leq 2^{\Gamma_p+1}$, and

$$\prod_{i=1}^{n} M(\hat{z}_i) \leq 4 \cdot \prod_{i=1}^{k} M(z_i)^{m_i} < \frac{4\,\mathrm{Mea}(p)}{|p_n|} \leq \frac{4\,\|p\|_2}{|p_n|} \leq \frac{4\,\|p\|}{|p_n|} < 8\lambda.$$

□

2.2.2 Clustering

After candidate approximations $\hat{z}_1, \ldots, \hat{z}_n$ are computed using a fixed precision parameter b, we perform a partitioning of these approximations into k clusters $C_1, \ldots, C_k$, where k is given as an input. The clustering is described in detail below. It works in phases. At the beginning of a phase, it chooses an unclustered approximation and uses it as the seed for the cluster formed in this phase. Ideally, each of the clusters corresponds to a distinct root of p. The clustering algorithm satisfies the following properties: (1) For $b < b_0$, the algorithm may or may not succeed in finding k clusters. (2) For $b \geq b_0$, the clustering always succeeds.

Whenever the clustering succeeds, the cluster C_i with seed $\tilde{z}_i$ is contained in the disk $D_i := \Delta(\tilde{z}_i, r_i)$, where $r_i \approx \min(\frac{1}{n^2}, \frac{\tilde{\sigma}_i}{256n^2})$, and $\tilde{\sigma}_i = \min_{j \neq i} |\tilde{z}_i - \tilde{z}_j|$. Furthermore, for $b \geq b_0$, D_i contains the root z_i (under suitable numbering) and exactly m_i many approximations.

For the clustering, we exploit the fact that an approximate factorization $\hat{p}(x) = p_n \prod_i (x - \hat{z}_i)$ of p with $\|p - \hat{p}\| \leq 2^{-b} \|p\|$ yields m_i approximations $\hat{z}$ of the root z_i with distance less than $2^{-b/(2m_i)}$ to z_i (at least, for $b \geq b_0$). Thus, in simple terms, we aim to determine clusters C of maximal size such that the pairwise distance between two elements in the same cluster is less than $2 \cdot 2^{-b/(2|C|)}$. We next give the details.

1. Initialize $\mathscr{C}$ to the empty set (of clusters).
2. Initialize C to the set of all unclustered approximations and choose $\hat{z} \in C$ arbitrarily. Let $a := 2^{\lfloor \log n \rfloor + 2}$ and $\delta := 2^{-b/4}$.
3. Update C to the set of points $q \in C$ satisfying $|\hat{z} - q| \leq 2\sqrt[a/2]{\delta}$.
4. If $|C| \geq a/2$, add C to $\mathscr{C}$. Otherwise, set $a := a/2$ and continue with step 3.
5. If there are still unclustered approximations, continue with step 2.
6. If the number of clusters in $\mathscr{C}$ is different from k, report failure, double b and go back to the factorization step.

Note that, for $b \geq b_0$, the disks $\Delta(z_i, 2^{-b/(2m_i)})$ are disjoint. Let Z_i denote the set of root approximations in $\Delta(z_i, 2^{-b/(2m_i)})$. Then, $|Z_i| = m_i$ according to Lemma 4. We show that, for $b \geq b_0$, the clustering algorithm terminates with $C = Z_i$ if called with an approximation $\hat{z} \in Z_i$.

Lemma 6. *Assume $b \geq b_0$, $\hat{z}_i \in Z_i$, $\hat{z}_j \in Z_j$, and $i \neq j$. Then,*

$$|\hat{z}_i - \hat{z}_j| \geq 2 \cdot (2^{-b/(16m_i)} + 2^{-b/(16m_j)}).$$

Proof. Since $b \geq b_0$, we have $2^{-b/(2m_\ell)} \leq \sigma_\ell$ for $\ell = i, j$ by (7) and $2^{-b/(16m_\ell)} = (2^{-b/(2m_\ell)})^{1/8} \leq \sigma_\ell/(4n) \leq \sigma_\ell/8$ by (9). Thus,

$$\begin{aligned} |\hat{z}_i - \hat{z}_j| &\geq \max(\sigma_i, \sigma_j) - 2^{-b/(2m_i)} - 2^{-b/(2m_j)} \\ &\geq \sigma_i/2 + \sigma_j/2 - \sigma_i/4 - \sigma_j/4 \\ &\geq 2 \cdot (2^{-b/(16m_i)} + 2^{-b/(16m_j)}). \end{aligned}$$

□

Lemma 7. *If $b \geq b_0$, the clustering algorithm computes the correct clustering, that is, it produces clusters C_1 to C_k such that $C_i = Z_i$ for all i (under suitable numbering). Let $\tilde{z}_i$ be the seed of C_i and let $\tilde{\sigma}_i = \min_{j \neq i} |\tilde{z}_i - \tilde{z}_j|$. Then, $(1-1/n)\sigma_i \leq \tilde{\sigma}_i \leq (1+1/n)\sigma_i$ and C_i as well as the root z_i is contained in $\Delta(\tilde{z}_i, \min(\frac{1}{n^2}, \frac{\tilde{\sigma}_i}{256n^2}))$.*

Proof. Assume that the algorithm has already produced Z_1 to Z_{i-1} and is now run with a seed $\hat{z} \in Z_i$. We prove that it terminates with $C = Z_i$. Let ℓ be a power of two such that $\ell \leq m_i < 2\ell$. The proof that the algorithm terminates with $C = Z_i$ consists of two parts. We first assume that steps 2 and 3 are executed for $a = 2\ell$. We show that the algorithm will then terminate with $C = Z_i$. In the second part of the proof, we show that the algorithm does not terminate as long as $a > 2\ell$.

Assume the algorithm reaches steps 2 and 3 with $a/2 = \ell$, i.e. $a/2 \leq m_i < a$. For any approximation $q \in Z_i$, we have $|\hat{z} - q| \leq 2 \cdot 2^{-b/(2m_i)} = 2\sqrt[m_i/2]{\delta} \leq 2\sqrt[a/2]{\delta}$. Thus, $Z_i \subseteq C$. Conversely, consider any approximation $q \notin Z_i$. Then, $|\hat{z} - q| \geq 2 \cdot 2^{-b/(16m_i)} > 2\sqrt[4m_i]{\delta} \geq 2\sqrt[2a]{\delta}$, and thus no such approximation is contained in C. This shows that $C = Z_i$. Since $|C| \geq a/2$, the algorithm terminates and returns Z_i.

It is left to argue that the algorithm does not terminate before $a/2 = \ell$. Since ℓ and a are powers of two, assume we terminate with $a/2 \geq 2\ell$, and let C be the cluster returned. Then, $m_i < a/2 \leq |C| < a$ and Z_i is a proper subset of C. Consider any approximation $q \in C \setminus Z_i$, say $q \in Z_j$ with $j \neq i$. Since $q \notin Z_i$, we have $|q - \hat{z}| \geq 2 \cdot (2^{-b/(16m_i)} + 2^{-b/(16m_j)}) > 2 \cdot 2^{-b/(16m_i)} > 2\sqrt[4m_i]{\delta}$. And since $q \in C$, we have $|q - \hat{z}| \leq 2\sqrt[a/2]{\delta}$. Thus, $4m_i \leq a/2$ and, hence, there are at least $3a/8$ many approximations in $C \setminus Z_i$. Furthermore, $|z_i - z_j| \leq |z_i - \hat{z}| + |\hat{z} - q| + |q - z_j| \leq 2^{-b/(2m_i)} + 2\sqrt[a/2]{\delta} + 2^{-b/(2m_j)} \leq 2^{-b/(16m_i)} + 2\sqrt[a/2]{\delta} + 2^{-b/(16m_j)} \leq 3\sqrt[a/2]{\delta}$. Consequently, there are at least $3a/8$ roots $z_j \neq z_i$ counted with multiplicity within distance $3\sqrt[a/2]{\delta}$ to z_i. This observation allows us to upper bound the value of $|P_i|$, namely

$$\begin{aligned} |P_i| &= \prod_{j \neq i} |z_i - z_j|^{m_j} \leq (3\sqrt[a/2]{\delta})^{3a/8} 2^{(n - m_i - 3a/8)\Gamma_p} \\ &< 3^n \delta^{3/4} 2^{n\Gamma_p} \leq 3^n 2^{-3b/16} 2^{n\Gamma_p} < 3^n 2^{-b/8} \cdot 2^{n\Gamma_p}, \end{aligned}$$

a contradiction to (10).

We now come to the claims about $\tilde{\sigma}_i$ and the disks defined in terms of it. The relation between σ_i and $\tilde{\sigma}_i$ follows from the second part of Lemma 4. All points in $C_i = Z_i$ have distance at most $2 \cdot 2^{-b/(2m_i)}$ from $\tilde{z}_i$. Also, by (6) and (9),

$$2 \cdot 2^{-b/(2m_i)} < \min(1/n^2, \sigma_i/(512n^2)) \leq \min(1/n^2, \tilde{\sigma}_i/(256n^2))$$

Hence, C_i as well as z_i is contained in $\Delta(\tilde{z}_i, \min(1/n^2, \tilde{\sigma}_i/(256n^2)))$. □

Lemma 8. *For a fixed precision b, computing a complete clustering needs $\tilde{O}(nb + n^2\Gamma_p)$ bit operations.*

Proof. For each approximation, we examine the number of distance computations we need to perform. Recall that b (property (5)) and a are powers of two, $a \leq 4n$ by definition, and $b \geq 8n \geq 2a$ by porperty (5). Then, $\sqrt[a/2]{\delta} = 2^{-b/(2a)} \in 2^{-\mathbb{N}}$. Thus, the number $\sqrt[a/2]{\delta}$ has a very simple format in binary notation. There is a single one, and this one is $b/(2a)$ positions after the binary point. In addition, all approximations $\hat{z}$ have absolute value less than $2 \cdot 2^{\Gamma}$ due to Step 5 in the overall algorithm. Thus, each evaluation of the form $|\hat{z} - q| \leq 2\sqrt[a/2]{\delta}$ can be done with

$$O(\Gamma + \log \delta^{-2/a}) = O((b/a) + \Gamma) = O((b/a) + \Gamma_p + \log n)$$

bit operations.

For a fixed seed $\hat{z}$, in the i-th iteration of step 2, we have at most $a \leq n/2^{i-2}$ many unclustered approximations left in C, since otherwise we would have terminated in an earlier iteration. Hence, we perform at most a evaluations of the form $|\hat{z} - q| \leq 2\sqrt[a/2]{\delta}$, resulting in an overall number of bit operations of $a \cdot O((b/a) + \Gamma) = O(b + a\Gamma)$ for a fixed iteration. As we halve a in each iteration, we have at most $\log n + 2$ iterations for a fixed $\hat{z}$, leading to a bit complexity of $O(b \log n + n\Gamma) = \tilde{O}(b + n\Gamma) = \tilde{O}(b + n\Gamma_p)$.

In total, performing a complete clustering has a bit complexity of at most $\tilde{O}(nb + n^2\Gamma_p)$. □

When the clustering succeeds, we have k clusters C_1 to C_k and corresponding seeds $\tilde{z}_1,\dots,\tilde{z}_k \subseteq \{\hat{z}_1,\dots,\hat{z}_n\}$. For $i = 1,\dots,k$, we define $D_i := \Delta(\tilde{z}_i, r_i)$, where $\tilde{z}_i$ is the seed for the cluster C_i and

$$r_i := \min(2^{-\lceil 2\log n\rceil}, 2^{\lceil \log \tilde{\sigma}_i/(256n^2)\rceil}) \geq \min(\frac{1}{2n^2}, \frac{\tilde{\sigma}_i}{256n^2}). \tag{11}$$

In particular, r_i is a 2-approximation of $\min(1/n^2, \tilde{\sigma}_i/(256n^2))$. Notice that the cost for computing the separations $\tilde{\sigma}_i$ is bounded by $\tilde{O}(nb + n^2\Gamma_p)$ bit operations since we can compute the nearest neighbor graph of the points $\tilde{z}_i$ (and thus the values $\tilde{\sigma}_i$) in $O(n\log n)$ steps [10] with a precision of $O(b + n\Gamma)$.

Now, suppose that $b \geq b_0$, Then, according to Lemma 7, the cluster C_i is contained in the disk D_i. Furthermore, D_i contains exactly one root z_i of p (under suitable numbering of the roots), and it holds that $m_i = \mathrm{mult}(z_i, p) = |C_i|$ and $\min(1/(2n^2), \sigma_i/(512n^2)) \leq r_i \leq \min(1/n^2, \sigma_i/(64n^2))$. If the clustering succeeds for a $b < b_0$, we have no guarantees (actually, the termination condition in step 4 gives some guarantee, however, we have chosen not to exploit it). Hence, before we proceed, we verify that each disk D_i actually contains the cluster C_i. If this is not the case, then we report a failure, return to the factorization step with $b = 2b$, and compute a new corresponding clustering.

In the next and final step, we aim to show that each of the enlarged disks $\Delta_i := \Delta(\tilde{z}_i, R_i) := \Delta(\tilde{z}_i, nr_i)$, $i = 1,\dots,k$, contains exactly one root z_i of p, and that the number of elements in $C_i \subseteq \Delta_i$ equals the multiplicity of z_i. Notice that, from the definition of r_i and Δ_i, it obvious that the disks Δ_i are pairwise disjoint and that $C_i \subseteq D_i \subseteq \Delta_i$.

2.2.3 Certification

In order to show that Δ_i contains exactly one root of p with multiplicity $|C_i|$, we show that each Δ_i contains the same number of roots of p and $\hat{p}$ counted with multiplicity. For the latter, we compute a lower bound for $|\hat{p}(z)|$ on the boundary $\mathrm{bd}\,\Delta_i$ of Δ_i, and check whether this bound is larger than $|(\hat{p}-p)(z)|$ for all points $z \in \mathrm{bd}\,\Delta_i$. If this is the case, then we are done according to Rouché's theorem. Otherwise, we start over the factorization algorithm with $b = 2b$. We now come to the details:

1. Let $\lambda = 2^{l_\lambda}$ be the 2-approximation of $\|p\|/|p_n|$ as defined in step 2 of the overall algorithm.
2. For $i = 1,\dots,k$, let $z_i^* := \tilde{z}_i + n \cdot r_i \in \Delta_i$. Note that $|z_i^*| \leq (1 + 1/n) \cdot M(\tilde{z}_i)$ since $nr_i \leq 1/n$.
3. We try to establish the inequality

$$|\hat{p}(z_i^*)/p_n| > E_i := 32 \cdot 2^{-b}\lambda M(\tilde{z}_i)^n \tag{12}$$

   for all i. We will see in the proof of Lemma 10 that this implies that each disk Δ_i contains exactly one root z_i of p and that its multiplicity equals the number $|C_i|$ of approximations within Δ_i. In order to establish the inequality, we consider $\rho = 1, 2, 4, 8, \dots$ and compute $|\hat{p}(z_i^*)/p_n|$ to an absolute error less than $2^{-\rho}$. If, for all $\rho \leq b$, we fail to show that $|p(z_i^*)/p_n| > E_i$, we report a failure and go back to the factorization algorithm with $b = 2b$. Otherwise, let ρ_i be the smallest ρ for which we are successful.
4. If, at any stage of the algorithm, $\sum_i \rho_i > b$, we also report a failure and go back to the factorization algorithm with $b = 2b$. Lemma 9 then shows that, for fixed b, the number of bit operations that are used for all evaluations is bounded by $\tilde{O}(nb + n^2\tau_p + n^3)$.
5. If we can verify that $|\hat{p}(\tilde{z}_i + nr_i)/p_n| > E_i$ for all i, we return the disks Δ_i and the multiplicities $m_i = |C_i|$.

Lemma 9. *For any i, we can compute $|\hat{p}(z_i^*)/p_n|$ to an absolute error less than $2^{-\rho}$ with a number of bit operations less than*

$$\tilde{O}(n(n + \rho + n\log M(\tilde{z}_i) + \tau_p)).$$

For a fixed b, the total cost for all evaluations in the above certification step is bounded by $\tilde{O}(nb + n^2\tau_p + n^3)$.

Proof. Consider an arbitrary subset $S \subseteq \{\hat{z}_1, \ldots, \hat{z}_n\}$. We first derive an upper bound for $\prod_{\hat{z}\in S} |z_i^* - \hat{z}|$. For that, consider the polynomial $\hat{p}_S(x) := \prod_{\hat{z}\in S}(x-\hat{z})$. The i-th coefficient of $\hat{p}_S$ is bounded by $\binom{|S|}{i} \cdot \prod_{\hat{z}\in S} M(\hat{z}) \le 2^n \prod_{i=1}^n M(\hat{z}_i) \le 8\lambda \cdot 2^n$ due to step 6 in the overall algorithm. It follows that

$$\begin{aligned}\prod_{\hat{z}\in S} |z_i^* - \hat{z}| = |\hat{p}_S(z_i^*)| &\le (n+1) M(z_i^*)^n \cdot 8\lambda \cdot 2^n \\ &< 64(n+1)^2 \cdot 2^n 2^{\tau_p} M(\tilde{z}_i)^n\end{aligned}$$

In order to evaluate $|\hat{p}(z_i^*)/p_n| = \prod_{j=1}^n |z_i^* - \hat{z}_j|$, we use approximate interval evaluation with an absolute precision $K = 1, 2, 4, 8, \ldots$. More precisely, we compute the distance of z_i^* to each of the points $\hat{z}_j$, $j = 1, \ldots, n$, up to an absolute error of 2^{-K}, and then take the product over all distances using a fixed point precision of K bits after the binary point.[6] We stop when the resulting interval has size less than $2^{-\rho}$. The above consideration shows that all intermediate results have at most $O(n + \tau_p + n \log M(\tilde{z}_i)$ bits before the binary point. Thus, we eventually succeed for an $K = O(\rho + \tau_p + n + n \log M(\tilde{z}_i))$. Since we have to perform n subtractions and n multiplications, the cost is bounded by $\tilde{O}(nK)$ bit operations for each K. Hence, the bound for the evaluation of $|\hat{p}(z_i^*)/p_n|$ follows.

We now come to the second claim. Since we double ρ in each iteration and consider at most $\log b$ iterations, the cost for the evaluation of $|\hat{p}(z_i^*)/p_n|$ are bounded by $\tilde{O}(n(n + \rho_i + n \log M(\tilde{z}_i) + \tau_p))$. Since we ensure that $\sum_i \rho_i \le b$, it follows that the total cost is bounded by $\tilde{O}(nb + n^2\tau_p + n^3 + n^2 \log(\prod_{i=1}^k M(\tilde{z}_i)))$. The last summand is smaller than $n^2 \cdot 8\lambda$ according to step 6, and $\lambda < 2\|p\|/|p_n| < 2(n+1)2^{\tau_p}$. This shows the claim. □

We now prove correctness of the certification algorithm. In particular, we show that the inequality (12) implies that the disk Δ_i contains the same number of roots of the polynomials $\hat{p}$ and p.

Lemma 10. *1. For all points $x \in \mathrm{bd}\,\Delta_i$, it holds that*

$$|\hat{p}(x)| \ge \frac{1}{4} |\hat{p}(z_i^*)| \,.$$

2. If inequality (12) holds for all i, then Δ_i isolates a root of z_i of p of multiplicity $m_i = \mathrm{mult}(z_i, p) = |C_i|$.

3. If $b \ge b_0$, then

$$\frac{|\hat{p}(z_i^*)|}{|p_n|} > \left(\frac{\min(256, \sigma_i)}{1024n}\right)^{m_i} \cdot \frac{|P_i|}{8} \ge 64 \cdot 2^{-b} \lambda M(\tilde{z}_i)^n$$

Proof. First, we show that, for any two points $x, y \in \mathrm{bd}\,\Delta_i$, their distance to any approximation $\hat{z} \in D_j$ differ only by a factor of $(1 + 1/n)$:

$$(1 - 1/n)|y - \hat{z}| \le |x - \hat{z}| \le (1 + 1/n)|y - \hat{z}| \,.$$

For a fixed $\hat{z}$, assume x to be the farthest point on $\mathrm{bd}\,\Delta$ from $\hat{z}$, and let y be the nearest. For $i \ne j$, we have

$$\begin{aligned}|x - \hat{z}| &\le |x - \tilde{z}_i| + |\tilde{z}_i - \tilde{z}_j| + |\tilde{z}_j - \hat{z}| \le (1 + 1/n)|\tilde{z}_i - \tilde{z}_j| \,, \text{and} \\ |y - \hat{z}| &\ge |\tilde{z}_i - \tilde{z}_j| - |y - \tilde{z}_i| - |\tilde{z}_j - \hat{z}| \ge (1 - 1/n)|\tilde{z}_i - \tilde{z}_j| \,.\end{aligned}$$

Similarly, for $i = j$:

$$\begin{aligned}|x - \hat{z}| &\le |x - \tilde{z}_i| + |\tilde{z}_i - \hat{z}| \le (1 + 1/n) n r_i \,, \text{and} \\ |y - \hat{z}| &\ge |y - \tilde{z}_i| - |\tilde{z}_i - \hat{z}| \ge (1 - 1/n) n r_i \,.\end{aligned}$$

Consequently, for any $x, y \in \mathrm{bd}\,\Delta_i$, it holds

$$|\hat{p}(x)| = |p_n| \cdot \prod_{\ell=1}^n |x - \hat{z}_\ell| \ge (1 + 1/n)^n |p_n| \cdot \prod_{\ell=1}^n |y - \hat{z}_\ell| \ge \frac{1}{4} |\hat{p}(y)| \,.$$

[6] In fact, we compute an interval I_j of size less than 2^{-K} such that $|z_i^* - \hat{z}_j| \in I_j$, and then consider the product $\prod_j I_j$.

This shows the first claim.

We turn to the second claim. Since $nr_i < 1/n$, we have $|x| < (1+1/n)M(\tilde{z}_i)$ for all $x \in \mathrm{bd}\Delta_i$. Now, if $|\hat{p}(z_i^*)/p_n| > 32 \cdot 2^{-b}\lambda M(\tilde{z}_i)^n$, then

$$\begin{aligned}|\hat{p}(x)| &> |\hat{p}(z_i^*)|/4 > 8|p_n|\lambda 2^{-b}(1-1/n)^n M(x)^n \\ &> \|p\| 2^{-b} M(x)^n \geq \|\hat{p}-p\| M(x)^n \geq |\hat{p}(x)-p(x)|.\end{aligned}$$

Hence, according to Rouché's theorem. Δ_i contains the same number (namely, $|C_i|$) of roots of p and $\hat{p}$. If this holds for all disks Δ_i, then each of the disks must contain exactly one root since p has k distinct roots. In addition, the multiplicity of each root equals the number $|C_i|$ of approximations within Δ_i.

It remains to show the third claim. Since $b \geq b_0$, it follows that $\min(1/(2n^2), \sigma_i/(512n^2)) \leq r_i \leq \min(1/n^2, \sigma_i/(64n^2))$ and $|\tilde{z}_i - z_i| < r_i$; cf. the remark following the definition of r_i in (11). Thus,

$$\begin{aligned}|\hat{p}(z_i^*)| &\geq |p(z_i^*)| - 2^{-b}\|p\| \cdot M(z_i^*)^n \\ &= |p(z_i + (\tilde{z}_i - z_i + nr_i))| - 2^{-b}\|p\| \cdot M(z_i^*)^n \\ &\geq ((n-1)r_i)^{m_i}|p_n P_i|/4 - 4 \cdot 2^{-b}\|p\| M(z_i)^n \\ &\geq \left(\frac{(n-1)\min(256,\sigma_i)}{512n^2}\right)^{m_i} |p_n P_i|/4 - 4 \cdot 2^{-b}\|p\| M(z_i)^n \\ &\geq \left(\frac{\min(256,\sigma_i)}{1024n}\right)^{m_i} |p_n P_i|/4 - 4 \cdot 2^{-b}\|p\| M(z_i)^n,\end{aligned}$$

where the first inequality is due to $|(p-\hat{p})(x)| < 2^{-b}\|p\| \cdot M(x)^n$, the second inequality follows from $|\tilde{z}_i - z_i + nr_i| \leq (n+1)r_i \leq \sigma_i/n$, Lemma 3 and $M(z_i^*) < (1+1/n) \cdot M(z_i)$, and the third inequality follows from $r_i \geq \min(\frac{1}{2n^2}, \frac{\sigma_i}{512n^2})$. In addition, we have

$$2^{-b}\|p\| M(z_i)^n \leq \left(\frac{\min(256,\sigma_i)}{1024n}\right)^{m_i} \cdot \frac{|p_n P_i|}{4096}, \tag{13}$$

since

$$\begin{aligned}&2^{-b}\|p\| M(z_i)^n \\ &\quad\leq 2^{-b/8} \cdot 2^{-b/2} \cdot 2^{\tau_p}|p_n| \cdot (n+1) \cdot M(z_i)^n \\ &\quad\leq \frac{|P_i|}{(n+1)2^{2n\Gamma_p+8n}} \left(\frac{\min(256,\sigma_i)}{1024n}\right)^{m_i} 2^{\tau_p}|p_n|\,(n+1)M(z_i)^n \\ &\quad\leq \left(\frac{\min(256,\sigma_i)}{1024n}\right)^{m_i} \cdot \frac{|p_n P_i|}{2^{7n-1}} \leq \left(\frac{\min(256,\sigma_i)}{1024n}\right)^{m_i} \cdot \frac{|p_n P_i|}{4096},\end{aligned}$$

where the second inequality follows from (10), (9), and (6) [7], and the third inequality follows from $\tau_p \leq n\Gamma_p + n + 1$ (Lemma 1) and $M(z_i)^n \leq 2^{n\Gamma_p}$. Finally,

$$\begin{aligned}\frac{|\hat{p}(z_i^*)|}{|p_n|} &> \left(\frac{\min(256,\sigma_i)}{1024n}\right)^{m_i} \cdot \frac{|P_i|}{8} \geq 512 \cdot 2^{-b}\frac{\|p\|}{|p_n|} M(z_i)^n \\ &\geq 64 \cdot 2^{-b}\lambda M(\tilde{z}_i)^n,\end{aligned}$$

where the first and the second inequality follow from (13) and the third inequality holds since λ is a 2-approximation of $\|p\|/|p_n|$ and $|z_i|^n \leq (1+1/n)^n|\tilde{z}_i|^n \leq 4|\tilde{z}_i|^n$. □

Lemma 11. *There exists a $b^* \geq b_0$ bounded by*

$$O\left(n\log n + n\Gamma_p + \sum_{i=1}^k \left(\log M(P_i^{-1}) + m_i \log M(\sigma_i^{-1})\right)\right)$$

[7] Observe $2^{-b/(2m_i)} \leq \min(\frac{1}{2n^2}, \frac{\sigma_i}{1024n^2}) \leq \frac{\min(256,\sigma_i)}{1024n}$.

such that the certification step succeeds for any $b > b^*$. The total cost in the certification algorithm (i.e. for all iterations until we eventually succeed) is bounded by

$$\tilde{O}\left(n^3 + n^2\tau_p + n \cdot \sum_{i=1}^{k} \left(\log M(P_i^{-1}) + m_i \log M(\sigma_i^{-1})\right)\right)$$

bit operations.

Proof. Due to Lemma 10,

$$|\hat{p}(z_i^*)/p_n| > \left(\frac{\min(256, \sigma_i)}{1024n}\right)^{m_i} \cdot \frac{|P_i|}{8} > 64 \cdot 2^{-b_0} \lambda M(\tilde{z}_i)^n.$$

Thus, in order to verify inequality (12), it suffices to evaluate $|\hat{p}(z_i^*)/p_n|$ to an error of less than $|\hat{p}(z_i^*)/2p_n|$. It follows that we succeed for some ρ_i with

$$\rho_i = O(m_i \log n + m_i \max(1, \log \sigma_i^{-1}) + \log \max(1, |P_i|^{-1})).$$

In Step 3 of the certification algorithm, we require that the sum over all ρ_i does not exceed b. Hence, we eventually succeed in verifying the inequality (12) for all i if b is larger than some b^* with

$$\begin{aligned} b^* &= O(b_0 + \textstyle\sum_i m_i \log n + \sum_i (\log M(P_i^{-1}) + m_i \log M(\sigma_i^{-1}))) \\ &= O(n \log n + n\Gamma_p + \textstyle\sum_i (\log M(P_i^{-1}) + m_i \log M(\sigma_i^{-1}))). \end{aligned}$$

For the bound for the overall cost, we remark that, for each b, the certification algorithm needs $\tilde{O}(n^3 + nb + n^2\tau_p)$ bit operations due to Lemma 9. Thus, the above bound follows from the fact that that we double b in each step and that the certification algorithm succeeds under guarantee for all $b > b^*$. □

2.3 Complexity of Root Isolation

We now turn to the complexity analysis of the root isolation algorithm. In the first step, we provide a bound for general polynomials p with real coefficients. In the second step, we give a simplified bound for the special case, where p has integer coefficients. We also give bounds for the number of bit operations that is needed to refine the isolating discs to a size less than $2^{-\kappa}$, with κ an arbitrary positive integer.

Theorem 3. *Let $p(x) \in \mathbb{C}[x]$ be a polynomial as defined in Section 2.1. We assume that the number k of distinct roots of p is given. Then, for all $i = 1, \ldots, k$, the algorithm from Section 2.2 returns an isolating disk $\Delta(\tilde{z}_i, R_i)$ for the root z_i together with the corresponding multiplicity m_i, and $R_i < \frac{\sigma_i}{64n}$.*

For that, it uses a number of bit operations bounded by

$$\tilde{O}\left(n^3 + n^2\tau_p + n \cdot \sum_{i=1}^{k} \left(\log M(P_i^{-1}) + m_i \log M(\sigma_i^{-1})\right)\right) \tag{14}$$

The algorithm needs an approximation of precision L of p, with L bounded by

$$O\left(n\Gamma_p + \sum_{i=1}^{k} \left(\log M(P_i^{-1}) + m_i \log M(\sigma_i^{-1})\right)\right). \tag{15}$$

Proof. For a fixed b, let us consider the cost for each of the steps in the algorithm:

- Steps 1-3, 5 and 6 do not use more than $\tilde{O}(n^2\Gamma_p + nb)$ bit operations,
- Step 4 and 7 do not use more than $\tilde{O}(n^2\Gamma_p + nb)$ bit operations (Corollary 1 and Lemma 8), and
- Step 8 and 9 use a number of bit operations bounded by (14) (Lemma 11).

In addition, for a fixed b, the oracle must provide an approximation of precision $O(n\Gamma_p + b)$ of p in order to compute the bound Γ for Γ_p, to compute the 2-approximation λ of $\|p\|/|p_n|$, and to run Pan's algorithm. The algorithm succeeds in computing isolating disks if $b > b^*$ with a b^* as in Lemma 11. Since

we double b in each step, we need at most $\log b^*$ iterations and the total cost for each iteration is bounded by (14). This shows the complexity result.

It remains to prove the bound for R_i. When the clustering succeeds, it returns disks $D_i = \Delta(\tilde{z}_i, r_i)$ with $\min(\frac{1}{2n^2}, \frac{\tilde{\sigma}_i}{256n^2}) \leq r_i \leq \min(\frac{1}{n^2}, \frac{\tilde{\sigma}_i}{128n^2})$ for all $i = 1, \ldots, m$. It follows that $R_i = n \cdot r_i \leq \frac{\tilde{\sigma}_i}{128n}$, and thus $|z_i - z_j| \geq |\tilde{z}_i - \tilde{z}_j| - |z_i - \tilde{z}_i| - |z_j - \tilde{z}_j| > |\tilde{z}_i - \tilde{z}_j| \cdot (1 - 1/(64n)) > |\tilde{z}_i - \tilde{z}_j|/2$ for all i, j with $i \neq j$. We conclude that $\sigma_i > \tilde{\sigma}_i/2 \geq 64nR_i$. □

We remark that the bound (14) can also be reformulated in terms of values that exclusively depend on the degree n and the geometry of the roots (i.e. their absolute values and their distances to each other). Namely, according to Lemma 1, we have $\tau_p \leq n + 1 + \log \frac{\mathrm{Mea}(p)}{|p_n|}$, and the latter expression only involves the degree and the absolute values of the roots of p. This yields the bound (2) from the introduction.

In the next step, we show that combining our algorithm with Pan's factorization algorithm also yields a very efficient method to further refine the isolating disks.

Theorem 4. *Let $p(x)$ be a polynomial as in Theorem 3, and κ be a given positive integer. We can compute isolating disks $\Delta_i(\tilde{z}_i, R_i)$ with radius $R_i < 2^{-\kappa}$ in a number of bit operations bounded by*

$$\mathscr{B} + \tilde{O}(n\kappa \cdot \max_{1 \leq i \leq k} m_i), \tag{16}$$

where $\mathscr{B}$ is bounded by (14). For that, we need an approximation of precision L of p with $L = \mathscr{L} + \tilde{O}(n\kappa \cdot \max_{1 \leq i \leq k} m_i)$, where $\mathscr{L}$ is bounded by (15).

Proof. As a first step, we use the algorithm from Section 2.2 to compute isolating disks $\Delta_i = \Delta(\tilde{z}_i, R_i)$ with $R_i \leq \sigma_i/(64n)$. Each disk Δ_i contains the root z_i, $m_i = \mathrm{mult}(z_i, p)$ approximations $\hat{z} \in \{\hat{z}_1, \ldots, \hat{z}_n\}$ of z_i, and it holds that $\sigma_i/2 < \tilde{\sigma}_i < 2\sigma_i$. Let

$$\hat{P}_i := \prod_{j : \hat{z}_j \notin \Delta_i} (\tilde{z}_i - \hat{z}_j).$$

We claim that $1/2|P_i| < |\hat{P}_i| < 2|P_i|$. Since $|\tilde{z}_i - z_i| < \sigma_i/(64n)$ for all i, it holds that $(1 - \frac{1}{64n})|z_i - z_j| \leq |\tilde{z}_i - \hat{z}| \leq (1 + \frac{1}{64n})|z_i - z_j|$ for all $j \neq i$ and $\hat{z} \in \Delta_j$. Thus, $|\hat{P}_i|$ is a 2-approximation of $|P_i|$. Similar as in the certification step, we now use approximate interval arithmetic to compute a 2-approximation μ_i of $|\hat{P}_i|$, and thus a 4-approximation of $|P_i|$. A completely similar argument as in the proofs of Lemma 9 and Lemma 11 then shows that we can compute such μ_i's with less than $\tilde{O}(n^3 + n^2\tau_p + n\sum_i \log M(P_i^{-1}))$ bit operations. Now, from the 2- and 4-approximations of σ_i and $|P_i|$, we can determine a b_κ such that

- the properties (5) to (8) are fulfilled, and
- $2^{-b/(2m_i)} < 2^{-\kappa}$.

Then, from Corollary 1 and Lemma 4, we conclude that Pan's factorization algorithm (if run with $b \geq b_\kappa$) returns, for all i, m_i approximations $\hat{z}$ of z_i with $|\hat{z} - z_i| < 2^{-b/(2m_i)} < 2^{-\kappa}$. Thus, for each i, we can simply choose an arbitrary approximation $\hat{z} \in \Delta_i$ and return the disk $\Delta(\hat{z}, 2^{-\kappa})$ which isolates z_i. The total cost splits into the cost for the initial root isolation and the cost for running Pan's Algorithm with $b = b_\kappa$. Since the latter cost is bounded by $\tilde{O}(nb_\kappa + n^2\Gamma_p)$, the bound (16) follows. □

Finally, we apply the above results to the important special case, where p is a polynomial with integer coefficients.

Theorem 5. *Let $p(x) \in \mathbb{Z}[x]$ be a polynomial of degree n with integer coefficients of size less than 2^τ. Then, we can compute isolating disks $\Delta(\tilde{z}_i, R_i)$, with $R_i < \frac{\sigma_i}{64n}$, for all roots z_i together with the corresponding multiplicities m_i using*

$$\tilde{O}(n^3 + n^2\tau) \tag{17}$$

bit operations. For a given positive integer κ, we can further refine the disks Δ_i to a size of less than $2^{-\kappa}$ with a number of bit operations bounded by

$$\tilde{O}(n^3 + n^2\tau + n\kappa). \tag{18}$$

Proof. In a first step, we compute the square-free part $p^* = p/\gcd(p,p')$ of p. According to [20, §11.2], we need $\tilde{O}(n^2\tau)$ bit operations for this step, and p^* has integer coefficients of bitsize $O(n+\tau)$. The degree of p^* yields the number k of distinct roots of p. Thus, we can directly apply our algorithm from Section 2.2 to the polynomial p. In order to derive the bound in (17), we have to reformulate the bound from (14) in terms of the degree n and the bitsize τ of p. From [8, Theorem 2], we conclude that $\sum_i m_i \log\max(1,\sigma_i^{-1}) = \tilde{O}(n^2 + n\tau)$. Furthermore, we have $\tau_p \le \tau$. Hence, it remains to show that $\sum_{i=1}^k \log M(P_i^{-1}) = \tilde{O}(n^3+n^2\tau)$. For that, we consider a square-free factorization $p(x) = \prod_{l=1}^n (Q_l(x))^l$ with square-free polynomials $Q_l \in \mathbb{Z}[x]$ such that Q_l and p/Q_l^l are coprime for all $l = 1,\dots,n$. Note that the roots of Q_l are exactly the roots of p with multiplicity l, and that Q_l is a constant for most l. We further denote $\bar{p} := p/\operatorname{lcf}(p)$ and $\bar{Q_l} := Q_l/\operatorname{lcf}(Q_l)$. Let S_l denote the set of roots of Q_l. Then, from the definition of P_i,

$$\begin{aligned}
\prod_{i:z_i\in S_l} |P_i| &= \prod_{i:z_i\in S_l}\prod_{j\neq i} |z_i - z_j|^{m_j} \\
&= \prod_{i:z_i\in S_l}\prod_{j\neq i:z_j\notin S_l} |z_i - z_j|^{m_j} \cdot \prod_{i:z_i\in S_l}\prod_{j\neq i:z_j\in S_l} |z_i - z_j|^l \\
&= \prod_{i\in S_l} |(\bar{p}/\bar{Q_l}^l)(z_i)| \cdot \prod_{i:z_i\in S_l} |(\bar{Q_l})'(z_i)|^l \\
&= |\operatorname{res}(\bar{p}/\bar{Q_l}^l, \bar{Q_l})| \cdot |\operatorname{res}(\bar{Q_l}, (\bar{Q_l})')|^l \\
&= \frac{|\operatorname{res}(p/Q_l^l, Q_l)|}{|\operatorname{lcf}(Q_l)^{n-l\cdot\deg Q_l}(\operatorname{lcf}(p/Q_l^l))^{\deg Q_l}|} \cdot \\
&\quad \cdot \left|\frac{\operatorname{res}(Q_l, Q_l')}{\operatorname{lcf}(Q_l)^{2\deg Q_l - 1}(\deg Q_l)^{\deg Q_l}}\right|^l \\
&\ge \frac{1}{|\operatorname{lcf}(Q_l)|^{n-l} \cdot |\operatorname{lcf}(p)|^{\deg Q_l} \cdot n^{l\deg Q_l}}
\end{aligned}$$

where $\operatorname{res}(f,g)$ denotes the resultant[8] of two polynomials f and g. For the last inequality, we used that $\operatorname{res}(p/Q_l^l, Q_l) \in \mathbb{Z}$ and $\operatorname{res}(Q_l, Q_l') \in \mathbb{Z}$. Taking the product over all l yields

$$\begin{aligned}
\prod_{i=1}^k |P_i| &\ge \left|\frac{1}{\prod_{l=1}^n (\operatorname{lcf}(Q_l)^{n-l} \cdot \operatorname{lcf}(p)^{\deg Q_l} \cdot n^{l\deg Q_l})}\right| \\
&\ge \frac{1}{|\operatorname{lcf}(p)|^{2n} \cdot n^n} \ge 2^{-2n\tau - n\log n}.
\end{aligned}$$

Note that, for any i, we also have

$$|P_i| = \frac{|p^{(m_i)}(z_i)|}{m_i! p_n} < \frac{m_i! 2^\tau (n+1) M(z_i)^n}{m_i! |p_n|} \le n 2^{\tau+1} M(z_i)^n,$$

and thus,

$$\sum_{i=1}^k \log M(P_i^{-1}) = \tilde{O}(n\tau + n\sum_{i=1}^k \log M(z_i)) = \tilde{O}(n\tau),$$

where we used that $\sum_i \log M(z_i)) \le \log \operatorname{Mea}(p) \le \log \|p\| < \log(n+1) + \tau$. This shows (17).

For the bound in (18) for the cost of refining the isolating disks $\Delta_i(\tilde{z}_i, R_i)$ to a size of less than $2^{-\kappa}$, we consider the square-free part p^*. Note that the disks Δ_i obtained in the first step are obviously also isolating for the roots of p^* (p and p^* have exactly the same distinct roots) and that $R_i < \sigma(z_i,p)/(64n) =$

[8] For univariate polynomials

$$\begin{aligned}
\operatorname{ref}(f,g) &= \operatorname{lcf}(f)^{\deg g} \operatorname{lcf}(g)^{\deg f} \prod_{(x,y):f(x)=g(y)=0} (x-y) \\
&= \operatorname{lcf}(f)^{\deg g} \prod_{x:f(x)=0} g(x).
\end{aligned}$$

$\sigma(z_i,p^*)/(64n) \le \sigma(z_i,p^*)/(64\deg p^*)$. Thus, proceeding in completely analogous manner as in the proof of Theorem 4 (with the square-free part p^* instead of p) shows that we need $\tilde{O}(n^3+n^2\tau+n\kappa)$ bit operations for the refinement. This proves the second claim. □

3 Curve Analysis

In this section, we show how to integrate our approach to certify roots of a univariate polynomial in an algorithm to compute a cylindrical algebraic decomposition [1, 11, 12, 19, 7, 3, 8, 14]. More specifically, we apply the results from the previous section to a recent algorithm, denoted TOPNT, from [3] for computing the topology of a real planar algebraic curve. This yields bounds on the expected number of bit operations for

- computing the topology of a real planar algebraic curve, and
- isolating the real solutions of a bivariate polynomial system

which improve the currently best bounds [8, 14] from $\tilde{O}(n^9\tau+n^8\tau^2)$ (deterministic) to $\tilde{O}(n^6+n^5\tau)$ (randomized) for topology computation and from $\tilde{O}(n^8+n^7\tau)$ (deterministic) to $\tilde{O}(n^6+n^5\tau)$ (randomized) for solving bivariate systems.

3.1 Review of the Algorithm TOPNT

The input of the algorithm is a bivariate polynomial $f \in \mathbb{Z}[x,y]$ of total degree n with integer coefficients of magnitude bounded by 2^τ. The polynomial defines an algebraic curve

$$C := \{(x,y) \in \mathbb{C}^2 : f(x,y) = 0\} \subseteq \mathbb{C}^2.$$

The algorithm returns a planar straight-line graph $\mathscr{G}$ embedded in $\mathbb{R}^2$ that is *isotopic*[9] *to the real part* $C_\mathbb{R} := C \cap \mathbb{R}^2$ of C.

In the first step (the **shearing step**), we choose an $s \in \mathbb{Z}$ at random[10] and consider the sheared curve

$$C_s := \{(x,y) \in \mathbb{C}^2 : f_s(x,y) := f(x+s\cdot y, y) = 0\}.$$

Then, any planar graph isotopic to the real part $C_{s,\mathbb{R}} := C_s \cap \mathbb{R}$ of C_s is also isotopic to $C_\mathbb{R}$, and vice versa. We choose s such that the leading coefficient (with respect to y) of the defining polynomial $f_s(x,y)$ of C_s is a constant. This guarantees that $C_{s,\mathbb{R}}$ has no vertical asymptote and that it contains no vertical line. By abuse of notation, we write $C = C_s$ and $f = f_s$ throughout the following considerations.

In the **projection step**, the x-critical points of C (i.e. all points $(x_0,y_0) \in C$ with $f_y(x_0,y_0) = 0$, $f_y := \frac{\partial f}{\partial y}$) are projected onto the real x-axis by means of a resultant computation. More precisely, we compute

- $R := \operatorname{res}(f, f_y; y) \in \mathbb{Z}[x]$,
- its square-free part $R^* := R/\gcd(R,R')$,
- isolating intervals $I_1,\ldots,I_k$ for the real roots $\alpha_1,\ldots,\alpha_k$ of R^*,
- the multiplicity $m_i := \operatorname{mult}(\alpha_i,R)$ of α_i as a root of R for all $i = 1,\ldots,m$, and
- arbitrary separating values $\beta_0,\ldots,\beta_{m+1} \in \mathbb{R}$ with $\alpha_k < \beta_{m+1}$, and $\beta_{i-1} < \alpha_i < \beta_i$ for all $i = 1,\ldots,m$.

We further compute

- $f_x^* := \frac{f_x}{\gcd(f_x,f_y)}$ and $f_y^* := \frac{f_y}{\gcd(f_x,f_y)}$,

[9] We actually consider the stronger notion of an *ambient isotopy*, but omit the "ambient". $\mathscr{G}$ is ambient isotopic to $C_\mathbb{R}$ if there is a continuous mapping $\phi : [0,1] \times \mathbb{R}^2 \mapsto \mathbb{R}^2$ with $\phi(0,\cdot) = \mathrm{id}_{\mathbb{R}^2}$, $\phi(1,C_\mathbb{R}) = \mathscr{G}$, and $\phi(t_0,\cdot)$ is a homeomorphism for each $t_0 \in [0,1]$.

[10] Initially, we consider $s = 0$.

- $Q := \mathrm{res}(f_x^*, f_y; y)$, and
- the multiplicity $l_i := \mathrm{mult}(\alpha_i, Q)$ of α_i as a root of Q for all $i = 1, \ldots, m$.

In the **lifting step**, we compute the fibers of C at the points α_i and β_i, that is, we isolate the roots of the polynomials $f_{\alpha_i}(y) := f(\alpha_i, y) \in \mathbb{R}[y]$ and $f_{\beta_i}(y) := f(\beta_i, y) \in \mathbb{R}[y]$. For that, we first compute the number of distinct complex roots of each of the latter polynomials, and then use the root isolator from Section 2.

Obviously, each polynomial $f_{\beta_i}(y)$ has $k(\beta_i) = \deg f_{\beta_i} = n$ distinct complex roots. The difficult part is to determine the number $k(\alpha)$ of distinct roots of $f_\alpha(y)$ for a root α of R^*. According to [3, (3.6)] and [3, Theorem 5], we have

$$k^+(\alpha) := n - \mathrm{mult}(\alpha, R) + \mathrm{mult}(\alpha, Q) \leq k(\alpha), \tag{19}$$

and, for a generic shearing factor s (i.e. for all but $n^{O(1)}$ many s), the equality $k^+(\alpha) = k(\alpha)$ holds for all roots α of R^*. Summation over all complex roots of R^* then yields

$$\begin{aligned} K^+ &:= \sum_{\alpha : R^*(\alpha) = 0} k^+(\alpha) = n \cdot \deg R^* - \deg R + \deg \gcd(R^\infty, Q) \\ &\geq \sum_{\alpha : R^*(\alpha) = 0} k(\alpha) =: K, \text{ and } K = K^+ \text{ for generic } s, \end{aligned}$$

where $\gcd(R^\infty, Q)$ is defined as the product of all common factors of R and Q with multiplicities according to their occurrence in Q. The crucial idea is now to compare the upper bound K^+ with a lower bound K^- which also equals K up to a non-generic choice of some parameters. In order to understand the computation of K^-, we first consider the exact computation of K: Let $\mathrm{Sres}_i(f, f_y; y) \in \mathbb{Z}[x, y]$ denote the *i-th subresultant polynomial* of f and f_y (with respect to y), and $\mathrm{sr}_i(x) := \mathrm{sres}_i(f, f_y; y) \in \mathbb{Z}[x]$ its leading coefficient. In particular, we have $R = \mathrm{sres}_0(f, g; y) = \mathrm{res}(f, f_y; y)$. We define:

$$\begin{aligned} &S_0 := R^*, && S_i := \gcd(S_{i-1}, \mathrm{sr}_i) \\ &R_1 := \frac{S_0}{S_1} = \frac{S_0}{\gcd(S_0, S_1)}, && R_i := \frac{S_{i-1}}{S_i} = \frac{\gcd(S_0, \ldots, S_{i-1})}{\gcd(S_0, \ldots, S_{i-1}, S_i)}, \end{aligned} \tag{20}$$

where $i = 1, \ldots, n$. Then, $\prod_{i \geq 1} R_i$ constitutes a factorization of R^* such that $R_i(\alpha) = 0$ if and only if $f(\alpha, y)$ has exactly $n - i$ distinct complex roots; see [3, Section 3.2.2] for details. Hence, we have $K = \sum_{i \geq 1} (n - i) \cdot \deg R_i$. Unfortunately, this computation of K is costly in practice due to the computation of the full subresultant sequence. Instead, in order to derive a lower bound K^-, we do not carry out the above computations directly in $\mathbb{Z}$ but in a modular prime field. More precisely, we choose a prime p at random, compute the modular images $\mathrm{sr}_i^{(p)}(x) = \mathrm{sres}_i(f \bmod p, f_y \bmod p; y) \in \mathbb{Z}_p[x]$ of $\mathrm{sr}_i(x) \in \mathbb{Z}[x]$, and perform all computations from (20) in $\mathbb{Z}_p[x]$. This yields polynomials $R_i^{(p)} \in \mathbb{Z}_p[x]$. Now, [3, Lemma 4] shows that

$$K^- := \sum_{i \geq 1} (n - i) \cdot \deg R_i^{(p)} \leq K, \tag{21}$$

and $K^- = K$ for all but finitely many bad primes.[11] Hence, if $K^- < K^+$, we have either chosen a bad prime or a bad shearing factor. In this case, we start over with a new s and choose a new prime p in the lifting step. If $K^- = K^+$, we know for sure that $K^+ = K$, and thus $k^+(\alpha) = k(\alpha)$ for all roots α of the resultant polynomial R.

We can now use our method from Section 2 to isolate all complex roots of the fiber polynomials $f_{\alpha_i}(y)$ and $f_{\beta_i}(y)$. Namely, we can ask for arbitrary good approximations of α_i and β_i (by refining corresponding isolating intervals), and thus for arbitrary good approximations of the coefficients of the fiber polynomials. In addition, we know the exact number of distinct roots of either polynomial. From the isolating regions in $\mathbb{C}$, we then derive isolating intervals for the real roots together with corresponding multiplicities. If one of the polynomials $f_{\alpha_i}(y)$ has more than one multiple real root, we start over and choose a new shearing factor s. Otherwise, we proceed with the final step.

[11] In the computation of the S_i's and R_i's (over $\mathbb{Z}$), all intermediate results have integer coefficients of bitsize bounded by $(n\tau)^{O(1)}$. Since the product of N distinct primes is larger than $N! = 2^{\Omega(N \log N)}$, there exist at most $(n\tau)^{O(1)}$ many bad primes for which $K^- \neq K$.

Connection step. We remark that, except for finitely many s, each f_{α_i} has exactly one multiple root. The previous two steps already yield the vertices of the graph $\mathscr{G}$. Namely, these are exactly the points[12]

$$V(\mathscr{G}) := \{(x,y) \in \mathbb{R}^2 : \exists i \text{ with } x = \alpha_i \text{ or } x = \beta_i, \text{ and } f(x,y) = 0\}$$

Since each polynomial $f_\alpha(y)$ has exactly one multiple root, there exists a unique vertex v along each vertical line, where either the number of edges connecting v to the left or to the right may differ from one. Hence, connecting all vertices in an appropriate manner is straightforward; see [3, Section 3.2.3] for more details.

3.2 Complexity Analysis

Throughout the following considerations, we say that a multivariate polynomial $G \in \mathbb{Z}[x_1,\ldots,x_k]$ with integer coefficients has *magnitude* (N,μ) if the total degree of G is bounded by N and all coefficients have absolute value of 2^μ or less. In addition, we fix the following notations: For an arbitrary $\alpha \in \mathbb{C}$,

- we define $f_\alpha(y) := \sum_{l=0}^{n} f_{\alpha,i} y^i := f(\alpha,y) \in \mathbb{C}[y]$, where $f \in \mathbb{Z}[x,y]$ is our input polynomial. We further define $\tau_\alpha := \log \max_i |f_{\alpha,i}| \geq 0$ as the logarithm of the maximal absolute value of the coefficients of f_α.
- the number of distinct roots of f_α is denoted by $k(\alpha)$. We further denote $z_{\alpha,1},\ldots,z_{\alpha,k(\alpha)}$ the distinct roots of f_α, and $m_{\alpha,i}$, $i = 1,\ldots,k(\alpha)$, the corresponding multiplicities.
- $\sigma_{\alpha,i}$ is the separation of $z_{\alpha,i}$, and $P_{\alpha,i} = \prod_{j\neq i}(z_{\alpha,i} - z_{\alpha,j})^{m_{\alpha,j}}$.
- For an arbitrary polynomial $G \in \mathbb{C}[x]$, we denote $V(G)$ the set of all distinct complex roots of G, and $\mathscr{V}(G)$ the multiset of all complex roots (i.e. each root occurs a number of times according to its multiplicity).

We first prove the following basic result:

Lemma 12. *For a fixed positive integer k, let $G \in \mathbb{Z}[x_1,\ldots,x_k]$ be an integer polynomial of magnitude (N,μ). Then, each divisor $g \in \mathbb{Z}[x_1,\ldots,x_k]$ of G has coefficients of bitsize $\tilde{O}(\mu+N)$.*

Proof. We prove the claim via induction over k. For a univariate $G \in \mathbb{Z}[x_1]$, we remark that $\mathrm{Mea}(g) \leq \mathrm{Mea}(G) \leq \|G\|_2 \leq 2^{\mu+1}N$, and thus the absolute value of each coefficient of g is bounded by $2^N \mathrm{Mea}(g) \leq 2^{N+\mu+1}N$.

For the general case, we write

$$g(x_1,\ldots,x_k) = \sum_{\lambda=(\lambda_1,\ldots,\lambda_{k-1})} a_\lambda(x_k) x_1^{\lambda_1}\cdots x_{k-1}^{\lambda_{k-1}}, \text{ with } a_\lambda \in \mathbb{Z}[x_k].$$

For a fixed $\bar{x_k} \in \{0,\ldots,N\}$, the polynomial $g(x_1,\ldots,x_{k-1},\bar{x_k}) \in \mathbb{Z}[x_1,\ldots,x_{k-1}]$ is a divisor of $G(x_1,\ldots,x_{k-1},\bar{x_k}) \in \mathbb{Z}[x_1,\ldots,x_{k-1}]$. Since $|\bar{x_k}|^N \leq N^N = 2^{N\log N}$ and $a_\lambda(x_k)$ has degree N or less, it follows that $G(x_1,\ldots,x_{k-1},\bar{x_k})$ has bitsize $O(N\log N+\mu)$. Hence, from the induction hypothesis, we conclude that the polynomial $g(x_1,\ldots,x_{k-1},\bar{x_k})$ has coefficients of bitsize $\tilde{O}(\mu+N)$, and thus $a_\lambda(i) \in \mathbb{Z}$ has bitsize $\tilde{O}(\mu+N)$ for all $i = 0,\ldots,N$ and all λ. Since a_λ is a polynomial of degree at most N, it follows that a_λ is uniquely determined by the values $a_\lambda(i)$, and thus Lagrange interpolation yields

$$a_\lambda(x) = \sum_{i=0}^{N} a_\lambda(i) \cdot \frac{x\cdot(x-1)\cdots(x-i+1)(x-i-1)\cdots(x-N)}{i\cdot(i-1)\cdots 1\cdot(-1)\cdots(i-N)}$$

Expanding the numerator of the fraction yields a polynomial with coefficients of absolute value $2^{O(N\log N)}$, and thus each coefficient of $a_\lambda(x_k)$ has bitsize $\tilde{O}(\mu+N)$ because $a_\lambda(i)$ has bitsize $\tilde{O}(\mu+N)$ and there are $N+1$ summands. This proves the claim. □

[12] For a graph with rational vertices, you may replace each $x_0 = \alpha_i$ (or $x_0 = \beta_i$) by an arbitrary rational value in its corresponding isolating interval, and the same for each real root of $f_{x_0}(y)$.

We now come to the complexity analysis for TOPNT. For the shearing step, we remark that there exist at most $n^{O(1)}$ many bad shearing factors s for which our algorithm does not succeed; see [3, Thm. 5] and [2, Prop. 11.23]. Thus, when choosing s at random, we can assume that we succeed for an integer s of bitsize $O(\log n)$. It follows that the sheared polynomial $f(x+sy,y)$ has magnitude $(n, O(\tau+\log n))$. Hence, throughout the following considerations, we can assume that the leading coefficient of f (with respect to y) is an integer constant and that f has magnitude $(n, O(\tau+\log n))$.

Lemma 13. *We can compute the entire subresultant sequence* $\mathrm{Sres}_i(f, f_y; y)$*, with* $i = 0, \ldots, n$*, the polynomial* $Q = \mathrm{res}(f_x^*, f_y^*; y)$*, and the square-free parts* R^* *and* Q^* *of the corresponding polynomials* $R = \mathrm{Sres}_0(f, f_y; y) = \mathrm{res}(f, f_y; y)$ *and* Q *with* $\tilde{O}(n^6 + n^5\tau)$ *bit operations.*

Proof. For two bivariate polynomials $g, h \in \mathbb{Z}[x,y]$ of magnitude (N, μ), computing the subresultant sequence $\mathrm{Sres}_i(g,h;y) \in \mathbb{Z}[x,y]$ together with the corresponding cofactor representations (i.e. the polynomials $u_i, v_i \in \mathbb{Z}[x,y]$ with $u_i g + v_i h = \mathrm{Sres}_i(g,h;y)$) needs $\tilde{O}(N^5\mu)$ bit operations [17, 5]. The total degree of the polynomials $\mathrm{Sres}_i(f, f_y; y)$ is bounded by N^2, the y-degree is bounded by $N-i$, and all coefficients have bitsize $\tilde{O}(N\mu)$. Furthermore, according to [20, §11.2], computing the square-free part of a univariate polynomial of magnitude (N, μ) uses $\tilde{O}(N^2\mu)$ bit operations, and the coefficients of the square-free part have bitsize $O(N+\mu)$. Hence, the claim concerning the computation of the polynomials $\mathrm{Sres}_i(f, f_y; y)$ and R^* follows from the fact that f and f_y have magnitude $(n, O(\tau+\log n))$ and R has magnitude $(n^2, \tilde{O}(n\tau))$.

From [2, Prop.10.14, Cor.10.15], we conclude that the polynomials f_x^* and f_y^* can directly be obtained as cofactors in the subresultant sequence of f_x and f_y. Thus, their computation needs $\tilde{O}(n^5\tau)$ bit operations as well. Since f_x^* divides f_x, and f_y^* divides f_y, Lemma 12 yields that f_x^* and f_y^* have magnitude $(n, \tilde{O}(n+\tau))$. Thus, for computing Q, we need $\tilde{O}(n^5\tau)$ bit operations, and Q has magnitude $(n^2, \tilde{O}(n^2+n\tau))$. Its square-free part Q^* can be computed using bit operations $\tilde{O}(n^5\tau + n^6)$. □

We now bound the cost for computing and comparing the roots of R and Q.

Lemma 14. *The roots of the polynomials* R *and* Q *can be computed with* $\tilde{O}(n^6 + n^5\tau)$ *bit operations. The same bound also applies to the number of bit operations that are needed to compute the multiplicities* $\mathrm{mult}(\alpha, R)$ *and* $\mathrm{mult}(\alpha, Q)$*, where* α *is a root of* R*.*

Proof. According to Theorem 5, we can compute isolating disks for the roots of the polynomials R and Q together with the corresponding multiplicities with $\tilde{O}(n^6 + n^5\tau)$ bit operations since R and Q have magnitude $(n^2, \tilde{O}(n^2+n\tau))$. For each root α of R, the algorithm returns a disk $\Delta^{(R)}(\alpha) := \Delta(\tilde{\alpha}, r_\alpha)$ with radius $r_\alpha < \frac{\sigma(\alpha,R)}{64 \deg R}$, and thus we can distinguish between real and non-real roots. A corresponding result also holds for each root β of Q, that is, each β is isolated by a disk $\Delta^{(Q)}(\beta)$ with radius less than $\frac{\sigma(\beta,Q)}{64 \deg Q}$. Furthermore, for any given positive integer κ, we can further refine all isolating disks to a size of less than $2^{-\kappa}$ with $\tilde{O}(n^6 + n^5\tau + n^2\kappa)$ bit operations.

For computing the multiplicities $\mathrm{mult}(\alpha, Q)$, where α is a root of R, we have to determine the common roots of R and Q. This can be achieved as follows: We first compute $d := \deg \gcd(R^*, Q^*)$ for which we need $\tilde{O}(n^6 + n^5\tau)$ bit operations. Namely, computing the gcd of two integer polynomials of magnitude (N, μ) needs $\tilde{O}(N^2\mu)$ bit operations. We conclude that R and Q have exactly d distinct roots in common. Hence, in the next step, we refine the isolating disks for R and Q until there are exactly d pairs $(\Delta^{(R)}(\alpha), \Delta^{(Q)}(\beta))$ of isolating disks that overlap. Since $P := R \cdot Q$ has magnitude $(2n^2, \tilde{O}(n^2+n\tau))$, the minimal distance between two distinct roots α and β is bounded by the separation of P, thus it is bounded by $2^{-\tilde{O}(n^4+n^3\tau)}$. We conclude that it suffices to refine the isolating disks to a size of $2^{-\tilde{O}(n^4+n^3\tau)}$, hence the cost for the refinement is again bounded by $\tilde{O}(n^6+n^5)$. Now, for each of the d pairs $(\Delta^{(R)}(\alpha), \Delta^{(Q)}(\beta))$ of overlapping disks, we must have $\alpha = \beta$, and these are exactly the common roots of R and Q. □

From the above Lemma, we conclude that we can compute the numbers $k^+(\alpha)$ for all roots α of R with $\tilde{O}(n^6 + n^5\tau)$ bit operations. Thus, the same bounds also applies to the computation of the upper bound $K^+ = \sum_\alpha k^+(\alpha)$ for $K = \sum_\alpha k(\alpha)$.[13] For the computation of the lower bound K^-, we have the following result:

[13] For simplicity, we ignored that (in practice) K^+ can be computed much faster from the equality $K^+ = n \cdot \deg R^* - \deg R + \deg \gcd(R^\infty, Q)$ instead of computing the $k^+(\alpha)$ first and, then, summing up all values.

Lemma 15. *We can compute K^- with $\tilde{O}(n^5\tau)$ bit operations.*

Proof. We have already computed the leading coefficients $\mathrm{sr}_i \in \mathbb{Z}[x]$ of the subresultant sequence $\mathrm{Sres}_i(f, f_y; y)$ and the square-free part $S_0 := R^*$ of the resultant polynomial R. Note that all polynomials S_i and R_i as defined in (20) have coefficients of bitsize $\tilde{O}(n\tau + n^2)$ because all of them divide R^*. Thus, except for $O(n\tau)$ many bad primes, the modular computation over $\mathbb{Z}_p$ yields polynomials $S_i^{(p)}, R_i^{(p)} \in \mathbb{Z}_p[x]$ with $\deg R_i^{(p)} = \deg R_i$ for all i, and thus $K^- = K$. Hence, we can assume that we only have to consider primes p of bitsize $O(\log(n\tau))$. Since we can compute the polynomials sr_i and R^* using $\tilde{O}(n^5\tau)$ bit operations, the same bound also applies to their modular computation over $\mathbb{Z}_p$.[14]

For the computation of the polynomials $S_i^{(p)} \in \mathbb{Z}_p[x]$, we have to perform at most n gcd computations (over $\mathbb{Z}_p$) involving polynomials of degree n^2. Thus, the cost for these computations is bounded by $\tilde{O}(n \cdot n^2 \log(n\tau))$ bit operations since computing the gcd of two polynomials in $Z_p[x]$ of degree N can be achieved with $\tilde{O}(N)$ arithmetic operations in $\mathbb{Z}_p$ due to [20, Prop. 11.6]. For the computation of the $R_i^{(p)}$'s, we have to consider the cost for at most n polynomial divisions. Again, for the latter computations, we need $\tilde{O}(n \cdot n^2 \log(n\tau))$ bit operations. □

We remark that it is even possible to compute K directly in an *expected* number of bit operations bounded by $\tilde{O}(n^5\tau)$. Namely, computing the gcd of two integer polynomials of magnitude (N, μ) needs an expected number of bit operations bounded by $\tilde{O}(N^2 + N\mu)$ according to [20, Prop. 11.11]. Hence, this yields the bound $\tilde{O}(n(n^4 + n^3\tau))$ for the expected number of bit operations to compute the polynomials S_i. Obviously, the same bound also applies to the computation of the R_i's.

For the analysis of the curve topology algorithm, it remains to bound the cost for isolating the roots of the "fiber polynomials" $f_{\alpha_i}(y) \in \mathbb{R}[x]$ and $f_{\beta_i}(y) \in \mathbb{R}[x]$, where the α_i's are the real roots of R and the β_i's are arbitrary separating values in between. In practice, we recommend to choose arbitrary rational values β_i, however, following this straight forward approach yields a bit complexity of $\tilde{O}(n^7 + n^6\tau)$ for isolating the roots of the polynomials $f_{\beta_i}(y) \in \mathbb{Q}[y]$. Namely, if β_i is a rational value with of L_i, then f_{β_i} has bitsize $\tilde{O}(nL_i + \tau)$. Thus, isolating the roots of f_{β_i} needs $\tilde{O}(n^3 L_i + n^2\tau)$ bit operations. However, since the separations of the α_i's are lower bounded by $2^{-\tilde{O}(n^4+n^3\tau)}$, we cannot get anything better than $\tilde{O}(n^4 + n^3\tau)$ for the largest L_i.

The crucial idea to improve upon the latter approach is to consider, for the values β_i, real roots of the polynomial $\hat{R}(x)$ instead, where $\hat{R} := \frac{(R^*)'}{\gcd((R^*)', (R^*)'')}$ is defined as the square-free part of the derivative of R^*. Note that the polynomials $\hat{R}$ and R do not share a common root, and, from the mean value theorem, we further conclude that, for any two consecutive real roots of R, there exists a root of $\hat{R}$ in between these two roots. We can obtain such separating roots by computing isolating disks for all complex roots of $\hat{R}$ such that none of these disks intersects any of the isolating disks for the roots of R. The computation of $\hat{R}$ needs $\tilde{O}(n^6 + n^5\tau)$ bit operations since $(R^*)'$ has magnitude $(n^2, \tilde{O}(n^2 + n\tau))$. We can use the same argument as in the proof of Lemma 14 to show that it suffices to compute isolating disks for R and $\hat{R}$ of size $2^{-\tilde{O}(n^4+n^3\tau)}$ in order to guarantee that the disks do not overlap. Again, Theorem 4 shows that we achieve this with $\tilde{O}(n^6 + n^5\tau)$ bit operations.

Now, throughout the following considerations, we assume that the separating elements β_i are real roots of $\hat{R}$ with $\beta_{i-1} < \alpha_i < \beta_i$. We will show in Lemma 19 that, for isolating the roots of all polynomials f_{β_i} and f_{α_i}, we need only $\tilde{O}(n^6 + n^5\tau)$ bit operations. For this purpose, we need the following result:

Lemma 16. *Let $G \in \mathbb{Z}[x]$ be a polynomial of magnitude (N, μ). For an arbitrary subset $V' \subset \mathscr{V}(G)$, it holds that*

$$\sum_{\alpha \in V'} \log \mathrm{Mea}(f_\alpha) = \tilde{O}(N\tau + n\mu), \text{ and } \quad \sum_{\alpha \in V'} \tau_\alpha = \tilde{O}(N\tau + n\mu).$$

In particular, for $G \in \{R, \hat{R}\}$, the bound write as $\tilde{O}(n^3 + n^2\tau)$.

[14] Note that, in practice, we never compute the entire subresultant sequence over $\mathbb{Z}$. In the proof of Lemma 13, we only assumed their exact computation in order to keep the argument simple and because of the fact that our overall complexity bound is not affected.

Proof. The proof is almost identical to the proof of Lemma 5 in [14]. The only difference is that we consider a general G, whereas in [14], only the case $G = R$ has been treated. Note that $\mathrm{Mea}_\alpha \geq 1$ for every $\alpha \in V(G)$, and that the Mahler measure is multiplicative, that means, $\mathrm{Mea}(g)\,\mathrm{Mea}(h) = \mathrm{Mea}(gh)$ for arbitrary univariate polynomials g and h. Therefore,

$$\sum_{\alpha\in V'} \log \mathrm{Mea}(f_\alpha) \leq \sum_{\alpha\in\mathscr{V}(G)} \log \mathrm{Mea}(f_\alpha) = \log \mathrm{Mea}\left(\prod_{\alpha\in\mathscr{V}(G)} f_\alpha\right).$$

Considering f as a polynomial in x with coefficients in $\mathbb{Z}[y]$ yields

$$\prod_{\alpha\in\mathscr{V}(G)} f_\alpha = \frac{\mathrm{res}(f,G;x)}{\mathrm{lcf}(G)^n} \Rightarrow \sum_{\alpha\in V'} \log \mathrm{Mea}(f_\alpha) \leq \log \mathrm{Mea}(\mathrm{res}(f,G;x)).$$

It is left to bound the degree and the bitsize of $\mathrm{res}(f,G;x)$. Considering the Sylvester matrix of f and G (whose determinant defines $\mathrm{res}(f,G;x)$), we observe that it has n rows with coefficients of G (which are integers of size $O(\mu)$) and N rows with coefficients of f (which are univariate polynomials of magnitude $(n,\tau+\log n)$). Therefore, the y-degree of $\mathrm{res}(f,G;y)$ is bounded by $O(nN)$, and its bitsize is bounded by $O(n(\mu+\log n)+N(\tau+\log n)) = \tilde{O}(N\tau+n\mu)$. This shows that $\log \mathrm{Mea}(\mathrm{res}(f,G;x)) = \tilde{O}(N\tau+n\mu)$, and thus the first claim follows.

For the second claim, note that the absolute value of each coefficient of $f_\alpha(y)$ is bounded by $(n+1)\cdot \lambda M(\alpha)^n$, where $\lambda = 2^{O(\tau+\log n)}$ is an upper bound for the absolute values of the coefficients of f. Thus, we have

$$\begin{aligned}\sum_{\alpha\in V'} \tau_\alpha &\leq \sum_{\alpha\in\mathscr{V}(G)} \tau_\alpha \leq \sum_{\alpha\in\mathscr{V}(G)} \log((n+1)\lambda M(\alpha)^n)\\ &= O(N(\tau+\log n)+n\log \mathrm{Mea}(G) = \tilde{O}(N\tau+n\mu)\end{aligned}$$

For the last claim, note that, for $G \in \{R,\hat{R}\}$, we have $N \leq n^2$ and $\mu = \tilde{O}(n^2+n\tau)$. □

Lemma 17. *For $G \in \{R,\hat{R}\}$, we have*

$$\sum_{\alpha\in V(G)} \sum_{i=1}^{k(\alpha)} m_{\alpha,i} \log M(\sigma_{\alpha,i}^{-1}) = \tilde{O}(n^4+n^3\tau)\ .$$

$$\sum_{\alpha\in V(G)} \sum_{i=1}^{k(\alpha)} \log M(P_{\alpha,i}^{-1}) = \tilde{O}(n^4+n^3\tau).$$

Proof. First, consider $G = R$. For any root α of R, we define $m(\alpha) := \mathrm{mult}(\alpha,R)$. From (19), we conclude that $\sum_{i=1}^{k(\alpha)}(m_{\alpha,i}-1) = n-k(\alpha) \leq n-k(\alpha)+\mathrm{mult}(Q,\alpha) \leq \mathrm{mult}(\alpha,R)$. Furthermore, since $m(\alpha)\geq 1$, it follows that $m_{\alpha,i} \leq m(\alpha)+1 \leq 2m(\alpha)$ for all i. Hence, we get

$$\begin{aligned}&\sum_{\alpha\in V(R)} \sum_{i=1}^{k(\alpha)} m_{\alpha,i} \log M(\sigma_{\alpha,i}^{-1}) \leq \sum_{\alpha\in V(R)} 2\cdot m(\alpha)\cdot \sum_{i=1}^{k(\alpha)} \log M(\sigma_{\alpha,i}^{-1})\\ &\overset{(1)}{=} \tilde{O}\left(\sum_{\alpha\in V(R)} m(\alpha)\cdot\left(n\log \mathrm{Mea}(f_\alpha)+\log M(\mathrm{sr}_{n-k(\alpha)}(\alpha)^{-1})\right)\right)\\ &\overset{(2)}{=} \tilde{O}\left(\sum_{\alpha\in\mathscr{V}(R)} n\log \mathrm{Mea}(f_\alpha)+\log M(\mathrm{sr}_{n-k(\alpha)}(\alpha)^{-1})\right)\\ &\overset{(3)}{=} \tilde{O}(n^4+n^3\tau).\end{aligned}$$

For (1), we used [14, 9] to show that

$$\sum_{i=1}^{k(\alpha)} \log M(\sigma_{\alpha,i}^{-1}) = \tilde{O}(n\log \mathrm{Mea}(f_\alpha)+\log M(\mathrm{sr}_{n-k(\alpha)}(\alpha)^{-1}))\ . \tag{22}$$

(2) follows from the fact that each $\alpha \in V(R)$ occurs $m(\alpha)$ times in $\mathscr{V}(R)$. Finally, for (3), we apply Lemma 16 to bound the first sum and [14, Lemma 8] to bound the second one.

The second claim can be shown as follows. For each α, we first split the sum

$$\sum_{i=1}^{k(\alpha)} \log M(P_{\alpha,i}^{-1}) = \sum_{i=1}^{k(\alpha)} \log |P_{\alpha,i}|^{-1} + \sum_{i;|P_{\alpha,i}|>1} \log |P_{\alpha,i}|. \tag{23}$$

Then, for the first sum, we have

$$\begin{aligned}
&\sum_{i=1}^{k(\alpha)} \log |P_{\alpha,i}|^{-1} = \sum_{i=1}^{k(\alpha)} \log \prod_{j\neq i} |z_{\alpha,i} - z_{\alpha,j}|^{-m_{\alpha,j}} \\
&= \log \left(\prod_{i=1}^{k(\alpha)} \prod_{j\neq i} |z_{\alpha,i} - z_{\alpha,j}| \right)^{-1} + \sum_{i=1}^{k(\alpha)} \log \prod_{j\neq i} |z_{\alpha,i} - z_{\alpha,j}|^{-(m_{\alpha,j}-1)} \\
&\overset{(1)}{\leq} \log \frac{|\mathrm{lcf}(f_\alpha)^{2k(\alpha)-2}| \cdot \prod_{i=1}^{k(\alpha)} m_{\alpha,i}}{|\mathrm{sr}_{n-k(\alpha)}(\alpha)|} + \sum_{i=1}^{k(\alpha)} \sum_{j\neq i} \log \sigma_{\alpha,i}^{-(m_{\alpha,j}-1)} \\
&\overset{(2)}{=} \tilde{O}(n\tau) + \log |\mathrm{sr}_{n-k(\alpha)}(\alpha)|^{-1} + \sum_{i=1}^{k(\alpha)} \log \sigma_{\alpha,i}^{-1} \sum_{j\neq i} (m_{\alpha,j} - 1) \\
&\overset{(3)}{\leq} \tilde{O}(n\tau) + \log |\mathrm{sr}_{n-k(\alpha)}|^{-1} + m(\alpha) \cdot \sum_{i=1}^{k(\alpha)} \log M(\sigma_{\alpha,i}^{-1}) \\
&\overset{(4)}{=} \tilde{O}(n\tau) + (m(\alpha)+1) \cdot \left(n \log \mathrm{Mea}\, f_\alpha + \log M(\mathrm{sr}_{n-k(\alpha)})^{-1} \right)
\end{aligned} \tag{24}$$

For (1), we have rewritten the product as a subresultant term, where we used [2, Prop. 4.28]. Furthermore, the distances $|z_{\alpha,i} - z_{\alpha,j}|$ have been lower bounded by the separation of $z_{\alpha,i}$. For (2), note that lcf_α is an integer of bitsize $O(\tau + \log n)$, that $k \leq n$, and that $\prod_i m_{\alpha,i} \leq n^n$. For (3), we used that $\sum_{j=1}^{k(\alpha)} (m_{\alpha,j} - 1) \leq m(\alpha)$, and, in (4), we applied (22). Now, summing up the expression in (24) over all $\alpha \in V(R)$ yields

$$\sum_{\alpha \in V(R)} \sum_{i=1}^{k(\alpha)} \log |P_{\alpha,i}|^{-1} = \tilde{O}(n^4 + n^3\tau),$$

where we again use Lemma 16 and [14, Lemma 8].

For the second sum in (23), we use that (cf. proof of Theorem 5)

$$|P_{\alpha,i}| = \frac{|f_\alpha^{(m_{\alpha,i})}(z_{\alpha,i})|}{m_{\alpha,i}! \mathrm{lcf}(f_\alpha)} < n 2^{\tau_\alpha + 1} M(z_{\alpha,i})^n,$$

and thus

$$\begin{aligned}
\sum_{\alpha \in V(R)} \sum_{i;|P_{\alpha,i}|>1} \log |P_{\alpha,i}| &= \tilde{O}\left(\sum_{\alpha \in V(R)} \left(n\tau_\alpha + n \log \prod_{i=1}^{k(\alpha)} M(z_{\alpha,i}) \right) \right) \\
&= \tilde{O}\left(\sum_{\alpha \in V(R)} (n\tau_\alpha + n \log \mathrm{Mea}(f_\alpha)) \right) \\
&= \tilde{O}(n^4 + n^3\tau)
\end{aligned}$$

according to Lemma 17. We conclude that

$$\sum_{\alpha \in V(R)} \sum_{i=1}^{k(\alpha)} \log M(P_{\alpha,i}^{-1}) = \tilde{O}(n^4 + n^3\tau).$$

Now, we consider the case $G = \hat{R}$. Note that, for each $\alpha \in V(\hat{R})$, we have $m_{\alpha,i} = 1$ for all i, and thus $k(\alpha) = n$. Namely, R and $\hat{R}$ do not share a common root, and thus each polynomial f_α has only simple roots. Also, $V(\hat{R}) = \mathscr{V}(\hat{R})$ since $\hat{R}$ is square-free. The following computation now shows the first claim

$$\begin{aligned}
&\sum_{\alpha\in V(\hat{R})} \sum_{i=1}^{k(\alpha)} m_{\alpha,i} \log M(\sigma_{\alpha,i}^{-1}) \\
&\quad = \sum_{\alpha\in V(\hat{R})} \left(\sum_{i=1}^{n} \log M(\sigma_{\alpha,i}^{-1}) \right) \\
&\quad = \tilde{O}\left(\sum_{\alpha\in V(\hat{R})} n \log \mathrm{Mea}(f_\alpha) + \log M(\mathrm{sr}_0(\alpha)^{-1}) \right) \\
&\quad = \tilde{O}\left(n^4 + n^3\tau + \sum_{\alpha\in V(\hat{R})} \log M(R(\alpha)^{-1}) \right).
\end{aligned}$$

In order to bound the sum in the above expression, note that

$$\sum_{\alpha\in V(\hat{R})} \log M(R(\alpha)^{-1}) = \sum_{\alpha\in V(\hat{R})} \log |R(\alpha)|^{-1} + \sum_{\alpha;|R(\alpha)|>1} |R(\alpha)|.$$

We first compute an upper bound for each value $|R(\alpha)|$. Since $R(x)$ has magnitude $(n^2, \tilde{O}(n\tau))$, it follows that $|R(\alpha)|$ has absolute value less than $2^{\tilde{O}(n\tau)} \cdot M(\alpha)^{n^2}$. Hence, for any subset $V' \subseteq V(\hat{R})$, it follows that

$$\sum_{\alpha\in V'} \log |R(\alpha)| \leq \tilde{O}(n^3\tau) + n^2 \log \mathrm{Mea}(\hat{R}) = \tilde{O}(n^4 + n^3\tau).$$

Thus, it is left to show that $\sum_\alpha \log |R(\alpha)|^{-1} = \tilde{O}(n^4 + n^3\tau)$, which follows from

$$\begin{aligned}
\sum_{\alpha\in V(\hat{R})} \log |R(\alpha)|^{-1} &= \log \prod_{\alpha\in V(\hat{R})} |R(\alpha)|^{-1} \\
&= \log \left(\frac{|\mathrm{lcf}(\hat{R})|^{\deg(R)}}{|\mathrm{res}(R,\hat{R})|} \right) = \tilde{O}(n^4 + n^3\tau)\ . \qquad (25)
\end{aligned}$$

In the second equation we rewrote the product in terms of the resultant $\mathrm{res}(R,\hat{R})$ [2, Prop. 4.16]. Since R and $\hat{R}$ have no common root, we have $|\mathrm{res}(R,\hat{R})| \geq 1$. Thus, the last equation follows from the fact that the leading coefficient of $\hat{R}$ has bitsize $\tilde{O}(n^2 + n\tau)$ and that $\deg(R) \leq n^2$.

Similarly, for the second claim, we first derive an upper bound for $\sum_{\alpha\in V(\hat{R})} \sum_{i:P_{\alpha,i}>1} \log |P_{\alpha,i}|$. Again, we can use exactly the same argument as for the case $G = R$ to show that the latter sum is bounded by $\tilde{O}(n^4 + n^3\tau)$. Hence, it suffices to prove that

$$\sum_{\alpha\in V(\hat{R})} \sum_{i=1}^{k(\alpha)} \log |P_{\alpha,i}|^{-1} = \tilde{O}(n^4 + n^3\tau).$$

This result follows from

$$\begin{aligned}
&\sum_{\alpha\in V(\hat{R})}\sum_{i=1}^{k(\alpha)}\log|P_{\alpha,i}|^{-1}\\
&\quad=\sum_{\alpha\in V(\hat{R})}\sum_{i=1}^{n}-\log\prod_{j\neq i}|z_{\alpha,i}-z_{\alpha,j}|\\
&\quad=\sum_{\alpha\in V(\hat{R})}\left(-\log\prod_{i=1}^{n}\prod_{j\neq i}|z_{\alpha,i}-z_{\alpha,j}|\right)\\
&\quad=\sum_{\alpha\in V(\hat{R})}\left(-\log\frac{|\mathrm{sr}_0(\alpha)|}{|\mathrm{lcf}(f_\alpha)^{2n-2}|}\right)\\
&\quad=\sum_{\alpha\in V(\hat{R})}\left(-\log\frac{|R(\alpha)|}{|\mathrm{lcf}(f_\alpha)^{2n-2}|}\right)=\tilde{O}(n^4+n^3\tau)\ .
\end{aligned}$$

The last step follows from (25). We remark that above computation is similar to the one for the case $G=R$. However, we used the fact that $\hat{R}$ is square-free, and thus all multiplicities $m_{\alpha,i}$ are equal to one. □

Lemma 18. *Let $G\in\{R,\hat{R}\}$ and let $L_\alpha\in\mathbb{N}$ be arbitrary positive integers, where α runs over all real roots of G. Then, we can compute approximations of precision L_α for all polynomials $f(\alpha,y)$ using*

$$\tilde{O}(n^6+n^5\tau+n^2\sum_\alpha L_\alpha)$$

bit operations.

Proof. For each α, we use approximate interval arithmetic to compute an approximation of the polynomial f_α. If we choose a fixed point precision ρ, and a starting interval of size $2^{-\rho}$ that contains α, then the so-obtained interval approximation of f_α has interval coefficients of size $2^{-\rho+2}(n+1)2^\tau M(\alpha)^n$; see again [13, Section 4] and [14, Section 5] for more details. Thus, in order to get an approximation of precision L_α of f_α, it suffices to consider a ρ of size $\tilde{O}(\tau+n\log M(\alpha)+L_\alpha)$. Thus, by doubling the precision ρ in each step, we eventually succeed for some $\rho=\rho_\alpha=\tilde{O}(\tau+n\log M(\alpha)+L_\alpha)$. The cost for the interval evaluations is then dominated (up to a logarithmic factor) by the cost in the last iteration. Thus, for a certain α, the cost is bounded by $\tilde{O}(n^2(\tau+n\log M(\alpha)+L_\alpha))$ since, for each of the $n+1$ coefficients of f_α, we have to (approximately) evaluate an integer polynomial (i.e. the coefficients of f considered as a polynomial in y) of magnitude $(n,O(\tau+\log n))$ at $x=\alpha$. The total cost for all α is then bounded by

$$\tilde{O}\left(n^2\cdot\sum_{\alpha\in\mathbb{R}\cap V(G)}\tau+n\log M(\alpha)+L_\alpha\right)=\tilde{O}(n^6+n^5\tau+n^2\sum_\alpha L_\alpha),$$

where we again used the result in Lemma 17. For the interval evaluations, we need an approximation of the root α to an absolute error of less than $2^{-\rho_\alpha}$. Such approximations are provided if we compute isolating disks of size less than $2^{-\kappa}$ for all roots of G, given that κ is larger than $\max_\alpha\rho_\alpha=\tilde{O}(\tau+n\max_\alpha\log M(\alpha)+\max_\alpha L_\alpha)=\tilde{O}(n^3+n^2\tau+\max_\alpha L_\alpha)$. In the proof of Lemma 14, we have already shown that we can compute such disks using $\tilde{O}(n^6+n^5\tau+n^2\kappa)$ bit operations. Thus, the claim follows. □

Lemma 19. *Let $G\in\{R,\hat{R}\}$. Then, computing isolating disks for all roots of all f_α, $\alpha\in V(R)\cap\mathbb{R}$, together with the corresponding multiplicities uses*

$$\tilde{O}(n^6+n^5\tau)$$

bit operations.

Proof. For a fixed α, let B_α be the number of bit operations that are needed to compute isolating disks for all roots of f_α together with the corresponding multiplicities. According to Theorem 3, we have

$$B_\alpha=\tilde{O}\left(n^3+n^2\tau_\alpha+n\cdot\textstyle\sum_{i=1}^{k(\alpha)}\left(m_{\alpha,i}\log M(\sigma_{\alpha,i}^{-1})+\log M(P_{\alpha,i}^{-1})\right)\right).$$

The corresponding algorithm from Section 2.2 returns isolating disks for the roots $z_{\alpha,i}$ and their mutiplicities $m_{\alpha,i}$. Furthermore, since the radius of the disk isolating $z_{\alpha,i}$ is smaller than $\sigma_{\alpha,i}/(64n)$, we can distinguish between real and non-real roots. The algorithm needs an approximation of precision L_α of f_α with

$$L_\alpha = \tilde{O}\left(n\tau_\alpha + \sum_{i=1}^{k(\alpha)} \left(m_{\alpha,i} \log M(\sigma_{\alpha,i}^{-1}) + \log M(P_{\alpha,i}^{-1})\right)\right).$$

From Lemma 18, we conclude that we can compute corresponding approximations for all f_α, $\alpha \in V(G) \cap \mathbb{R}$, with a number of bit operations bounded by

$$\tilde{O}(n^6 + n^5\tau + n^2 \cdot \sum_{\alpha \in V(G)\cap\mathbb{R}} L_\alpha).$$

The above expression is bounded by $\tilde{O}(n^6 + n^5\tau)$ because

$$\sum_{\alpha \in V(G)} \left(n\tau_\alpha + \sum_{i=1}^{k(\alpha)} \left(m_{\alpha,i} \log M(\sigma_{\alpha,i}^{-1}) + \log M(P_{\alpha,i}^{-1})\right)\right)$$

is bounded by $\tilde{O}(n^4 + n^3\tau)$ according to Lemma 16 and 17. The same argument also shows that the sum over all B_α is even bounded by $\tilde{O}(n^5 + n^4\tau)$. Hence, the claim follows. □

We can now formulate our main theorem:

Theorem 6. *Computing the topology of a real planar algebraic curve $C = V(f)$, where $f \in \mathbb{Z}[x,y]$ is a bivariate polynomial of total degree of n with integer coefficients of magnitude bounded by 2^τ, needs an expected number of bit operations bounded by*

$$\tilde{O}(n^6 + n^5\tau).$$

Proof. We already derived a bound of $\tilde{O}(n^6 + n^5\tau)$ or better for each of the steps in the projection and in the lifting phase of our algorithm. The final connection phase is purely combinatorial since we ensure that each f_α, with α a root of the resultant R, has at most one multiple real root. Thus, we can compute all adjacencies in linear time with respect to the number of roots of critical and intermediate fiber polynomials. Since their number is bounded by $O(n^3)$, this step can be done in $O(n^3)$ operations. □

Finally, obtain the following result for solving bivariate polynomial systems:

Theorem 7. *Let $g,h \in \mathbb{Z}[x,y]$ be coprime polynomials of magnitude (n,τ). Then, we can compute isolating boxes for the real solutions of the system $g(x,y) = h(x,y) = 0$ with an expected number of bit operations bounded by*

$$\tilde{O}(n^6 + n^5\tau).$$

Proof. The idea is to consider the polynomial $f(x,y) := g^2 + h^2$ and to compute the topology of the curve $C := C_\mathbb{R}$ defined by f. Since g and h are assumed to be coprime, the system $g = h = 0$ has only finitely many solutions, and the set of these points coincides with the "curve" C. Hence, the topology algorithm returns a graph that consists of vertices only. According to Theorem 6, the cost the topology computation is bounded by $\tilde{O}(n^6 + n^5\tau)$ bit operations in expectation since f has magnitude $(2n, O(\tau))$.

However, in general, our algorithm does not directly return the solutions of the initial system but the solutions of a sheared system $g(x+sy,y) = h(x+sy,y) = 0$. Here, s is a positive integer of bitsize $O(\log n)$ for which TOPNT succeeds in computing the topology of the sheared curve $\hat{C} := C_{s,\mathbb{R}}$ defined by $\hat{f}(x,y) = f(x+sy,y) = 0$. Since $\hat{C}$ consists of isolated singular points only and there are no two covertical points (note that our algorithm only succeeds for an s for which there are no two covertical extremal points), it follows that, for each point $(\hat{x},\hat{y}) \in \hat{C}$, $\hat{x}$ is a root of the resultant $\hat{R} = \mathrm{res}(\hat{f}, \hat{f}; y)$ and $\hat{y}$ is the unique (multiple) real root of $\hat{f}(\hat{x},y)$. The point $(\hat{x},\hat{y})$ is represented by an isolating box $B(\hat{x},\hat{y}) = I(\hat{x}) \times I(\hat{y})$, where $I(\hat{x})$ is the isolating interval for the root $\hat{x}$ of $\hat{R}$ and $I(\hat{y})$ is the isolating interval for the root $\hat{y}$ of $f(\hat{x},y)$. Each solution (x,y) of the initial system can now be recovered from a unique solution $(\hat{x},\hat{y}) \in \hat{C}$. More precisely, $x = \hat{x} - s \cdot \hat{y}$ and $y = \hat{y}$. However, in order to obtain isolating boxes for the solutions (x,y), we have to refine the boxes $B(\hat{x},\hat{y})$ first such that the sheared boxes $B(x,y) := (I(\hat{x}) - s \cdot I(\hat{y}), I(\hat{y}))$ do not overlap. Note

that the latter is guaranteed if both intervals $I(\hat{x})$ and $I(\hat{y})$ have width less than $\sigma(\hat{x},\hat{R})/(4|s|) \leq \sigma(\hat{x},\hat{R})/4$. Namely, if the latter inequality holds, then the intervals $I(\hat{x}) - s \cdot I(\hat{y})$ are pairwise disjoint. Hence, it follows that the corresponding isolating intervals have to be refined to a width less than $w(\hat{x},\hat{y}) = \sigma(\hat{x},\hat{R})/n^{O(1)}$. For the resultant polynomial $\hat{R}$, we conclude from Theorem 5 that computing isolating intervals of size less $w(\hat{x},\hat{y})$ uses $\tilde{O}(n^6 + n^5\tau)$ bit operations since $\log M(w(\hat{x},\hat{y})^{-1}) = \tilde{O}(n^4 + n^3\tau)$ and $\hat{R}$ has magnitude $(n^2, \tilde{O}(n\tau))$. In order to compute an isolating interval of size $w(\hat{x},\hat{y})$ or less for the root $\hat{y}$ of $\hat{f}(\hat{x},y)$ (in fact, for all roots of $\hat{f}(\hat{x},y)$), we need

$$\tilde{O}(n^3 + n^2\tau_{\hat{x}} + n \cdot \sum_{i=1}^{k(\hat{x})} \left(m_{\hat{x},i} \log M(\sigma_{\hat{x},i}^{-1}) + \log M(P_{\hat{x},i}^{-1}) \right) + (n \max_i m_{\hat{x},i}) \cdot \log M(w(\hat{x},\hat{y})^{-1})$$

bit operations; cf. the proof of Lemma 19 with $\alpha = \hat{x}$ and $f = \hat{f}$. Also, we need an approximation of precision $L_{\hat{x}}$ of $\hat{f}(\hat{x},y)$ with $L_{\hat{x}}$ bounded by

$$\tilde{O}(n\tau_{\hat{x}} + \sum_{i=1}^{k(\hat{x})} \left(m_{\hat{x},i} \log M(\sigma_{\hat{x},i}^{-1}) + \log M(P_{\hat{x},i}^{-1}) \right) + (n \max_i m_{\hat{x},i}) \cdot \log M(w(\hat{x},\hat{y})^{-1}).$$

Since $\max_i m_{\hat{x},i} \leq 2 \cdot \mathrm{mult}(\hat{x},\hat{R})$ and $w(\hat{x},\hat{y}) = \sigma(\hat{x},\hat{R})/n^{O(1)}$, it holds $(n \max_i m_{\hat{x},i}) \cdot \log M(w(\hat{x},\hat{y})^{-1}) = \tilde{O}(n \cdot \mathrm{mult}(\hat{x},\hat{R}) \cdot \log M(\sigma(\hat{x},\hat{R})^{-1})$. Thus, summing up the cost for computing the roots of $\hat{f}(\hat{x},y)$ over all real roots of $\hat{R}$ yields the bound $\tilde{O}(n^5 + n^4\tau)$. Here, we use an analogous argument as in the proof of Lemma 19 and the fact that $\sum_{\hat{x}} n \cdot \mathrm{mult}(\hat{x},\hat{R}) \cdot \log M(\sigma(\hat{x},\hat{R})^{-1} = \tilde{O}(n^5 + n^4\tau)$. The more costly part is to compute the approximations of precision $L_{\hat{x}}$ of the polynomials $\hat{f}(\hat{x},y)$. Again, we can use Lemma 16 and 17 to show that $\sum_{\hat{x}} L_{\hat{x}} = \tilde{O}(n^4 + n^3\tau)$. Thus, from Lemma 18, we conclude that the approximations of the $\hat{f}(\hat{x},y)$'s can be computed with $\tilde{O}(n^6 + n^5\tau)$ bit operations. □

References

[1] D. S. Arnon, G. E. Collins, and S. McCallum. Cylindrical Algebraic Decomposition I. *SIAM Journal of Computing*, 13(4):865–889, 1984.

[2] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*. Springer, 2nd edition, 2006.

[3] E. Berberich, P. Emeliyanenko, A. Kobel, and M. Sagraloff. Exact Symbolic-Numeric Computation of Planar Algebraic Curves. *CoRR*, abs/1201.1548, 2012.

[4] D. Bini and G. Fiorentino. Design, Analysis, and Implementation of a Multiprecision Polynomial Rootfinder. *Numerical Algorithms*, 23:127–173, 2000.

[5] D. I. Diochnos, I. Z. Emiris, and E. P. Tsigaridas. On the Asymptotic and Practical Complexity of Solving Bivariate Systems Over the Reals. *J. Symb. Comput.*, 44(7):818–835, 2009.

[6] A. Eigenwillig. *Real Root Isolation for Exact and Approximate Polynomials Using Descartes' Rule of Signs*. PhD thesis, Saarland University, Germany, 2008.

[7] A. Eigenwillig, M. Kerber, and N. Wolpert. Fast and Exact Analysis of Real Algebraic Plane Curves. In *ISSAC*, pages 151–158, New York, NY, USA, 2007. ACM.

[8] P. Emeliyanenko and M. Sagraloff. On the Complexity of Solving a Bivariate Polynomial System. In *ISSAC*, pages 154–161, New York, NY, USA, 2012. ACM.

[9] I. Z. Emiris, V. Y. Pan, and E. P. Tsigaridas. Algebraic Algorithms. available at `tr.cs.gc.cuny.edu/tr/files/TR-2012001.pdf`, 2012.

[10] D. Eppstein, M. Paterson, and F. Yao. On Nearest-Neighbor Graphs. *Discrete & Comput. Geometry*, 17:263–282, 1997.

[11] L. Gonzalez-Vega and M. E. Kahoui. An Improved Upper Complexity Bound for the Topology Computation of a Real Algebraic Plane Curve. *J. Complexity*, 12(4):527–544, 1996.

[12] H. Hong. An Efficient Method for Analyzing the Topology of Plane Real Algebraic Curves. *Mathematics and Computers in Simulation*, 42(4-6):571–582, 1996.

[13] M. Kerber and M. Sagraloff. Efficient Real Root Approximation. In *ISSAC*, pages 209–216, New York, NY, USA, 2011. ACM.

[14] M. Kerber and M. Sagraloff. A Worst-case Bound for Topology Computation of Algebraic Curves. *J. Symb. Comput.*, 47(3):239–258, 2012.

[15] K. Mehlhorn and M. Sagraloff. A Deterministic Descartes Algorithm for Real Polynomials. *J. Symb. Comput.*, 46(1):70 – 90, 2011. A preliminary version appeared in ISSAC 2009.

[16] V. Pan. Univariate Polynomials: Nearly Optimal Algorithms for Numerical Factorization and Root Finding. *J. Symb. Comput.*, 33(5):701–733, 2002.

[17] D. Reischert. Asymptotically Fast Computation of Subresultants. In *ISSAC*, pages 233–240, New York, NY, USA, 1997. ACM.

[18] A. Schönhage. The Fundamental Theorem of Algebra in Terms of Computational Complexity. Technical report, Math. Inst. Univ. Tübingen, 1982.

[19] A. Strzebonski. Cylindrical Algebraic Decomposition Using Validated Numerics. *J. Symb. Comp*, 41:1021–1038, 2006.

[20] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, Cambridge, 1999.

[21] C. K. Yap and M. Sagraloff. A Simple but Exact and Efficient Algorithm for Complex Root Isolation. In *ISSAC*, pages 353–360, New York, NY, USA, 2011. ACM.